*To appear in Proc. of **DL'96***

Representing SGML Documents in Description Logics

Diego Calvanese and **Giuseppe De Giacomo** and **Maurizio Lenzerini**
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, I-00198 Roma, Italy
{calvanese,degiacomo,lenzerini}@dis.uniroma1.it

Abstract

Recent proposals to improve the quality of interaction with the World Wide Web suggest considering the Web as a huge semistructured database, so that retrieving information can be supported by the task of database querying. Under this view, it is important to represent the form of both the network, and the documents placed in the nodes of the network. However, the current proposals do not pay sufficient attention to represent document structures and reasoning about them. In this paper, we address these problems by providing a framework where Document Type Definitions (DTDs) expressed in the Standard Generalized Markup Language are formalized in an expressive Description Logic equipped with sound and complete inference algorithms. We provide a method for verifying structural equivalence of DTDs, which works in worst case deterministic exponential time, in contrast to the known algorithms for this problem which are double exponential. We also deal with parametric versions of structural equivalence, and investigate other forms of reasoning on DTDs. The reasoning services studied in this paper can be seen as the fundamental building blocks for devising more complex inference systems supporting the task of querying the World Wide Web.

1 Introduction

The view of the World Wide Web as a large information system constituted by a collection of documents connected by hypertext links, is stimulating many lines of research related to database and knowledge representation issues. One of the most interesting aspects addressed in recent papers is the design of suitable declarative mechanisms for querying the Web information system making use of a representation of the structure of the network (see for example [Mendelzon *et al.*, 1996; Konopnicki and Shmueli, 1995; Lakshmanan *et al.*, 1996]). These works point out that taking into account the structure of documents placed in the nodes of the Web, and being able to reason on it, would help in several tasks related to query processing, such as query formulation, optimization and restructuring (see [Christophides *et al.*, 1994; Quass *et al.*, 1995; Mendelzon *et al.*, 1996]).

One common way of describing the structure of documents is by marking them up using tags. For example, HTML documents are defined in this way. The standard way of describing marked-up documents is by using *Document Type Definitions* (DTDs) expressed in *Standard Generalized Markup Language* (SGML). Given two DTDs, a natural and fundamental question is whether they are equivalent in the sense that they define the same sets of marked-up documents (strong equivalence). Other forms of reasoning on DTDs are also of practical interest [Wood, 1995; Raymond *et al.*, 1995], such as verifying structural equivalence, which is a weaker form of equivalence abstracting from tag names in the documents. SGML literature has stressed the need for reasoning methods to determine such equivalences, and for studying their computational properties [Wood, 1995].

In this paper, we show a formalization of SGML DTDs in a decidable Description Logic, which provides us with a general method to reason on DTDs. In this logic non-first-order constructs as reflexive and transitive closure and well-foundedness play a crucial role. The proposed formalization allows us to verify structural equivalence of DTDs in worst case deterministic exponential time, while the known algorithms for this problem are double exponential. We also introduce and study parametric versions of structural equivalence, and investigate other forms of reasoning on DTDs. The reasoning services studied in this paper can be seen as the fundamental building blocks for devising more complex inference systems supporting the task of querying the Web.

2 SGML Document Type Definitions

In this section we present Document Type Definitions which are used in SGML [Int, 1986] in order to define marked-up documents (Section 2.1), and various forms of reasoning on such documents (Section 2.2).

2.1 DTDs and Documents

An SGML document consists of an SGML prologue and a marked-up document instance. The prologue includes a *Document Type Definition (DTD)*, which is constituted by a set of *element type definitions* defining the generic structure of the various components of the marked-up document instance. The logical components of a document are called *elements.*

It is well known that the fundamental characteristics of DTDs can be formalized by means of Extended Context Free Grammars (ECFGs) [Wood, 1995]. Marked-up document instances are seen as syntax trees constructed according to the grammar, where the tree structure is determined by the various *tags* that occur in the document and that constitute the markup. An ECFG is a tuple $(\mathbf{E}, \mathbf{T}, \mathbf{P}, I)$, where $\mathbf{E}$ is an alphabet of *nonterminal symbols*, $\mathbf{T}$ is an alphabet of *terminal symbols*, $\mathbf{P}$ is a set of *production rules*, and $I \in \mathbf{E}$ is the initial symbol of the grammar. The nonterminal symbols are the elements defined in the DTD, and the start symbol is the element that specifies the document type. The terminal symbols are the basic types of SGML, such as `#PCDATA`, which represent generic (unmarked) strings with no associated structure within the DTD. In the following, with the term *symbol*, denoted by the letter S, we mean a generic terminal or nonterminal symbol in $\mathbf{E} \cup \mathbf{T}$. Each production rule $E \to \alpha$ of the ECFG corresponds to an element type definition. E is the defined element, and α, called *content model*, is an expression over the symbols of the grammar constructed according to the following syntax:

$$\alpha ::= S \mid \varepsilon \mid \alpha_1, \alpha_2 \mid \alpha_1|\alpha_2 \mid \alpha^*.$$

In fact, α is a regular expression with "," denoting concatenation and "|" denoting union. Additionally, in content models, the following standard abbreviations are used:

$$\begin{aligned} \alpha_1 \& \alpha_2 &= (\alpha_1, \alpha_2)|(\alpha_2, \alpha_1) \\ \alpha? &= \varepsilon|\alpha \\ \alpha^+ &= \alpha, \alpha^*. \end{aligned}$$

When no ambiguity may arise, we identify α with the set of words generated by the regular expression that α represents.

Figure 1 shows an example of a DTD M for a simple mail document, expressed in SGML syntax. It is straightforward to derive the set of ECFG productions corresponding to the various element type definitions.

DTDs contain in fact also other aspects that are not directly related to the document structure. An example is the possibility to associate to each element a set of properties by means of a so called *attribute list.* In the following, for the sake of simplicity, we do not consider those additional aspects. We remark, however, that the representation of DTDs in terms of Description Logics provided in Section 3, makes it straightforward to take also these aspects into consideration.

Let $\mathcal{D} = (\mathbf{E}, \mathbf{T}, \mathbf{P}, I)$ be a DTD. We assume without loss of generality that for each element $E \in \mathbf{E}$, $\mathbf{P}$ contains at most one element type definition $E \to \alpha$ where E appears on the left hand side. We also assume that for each element E appearing in $\mathbf{P}$, there is an element type definition $E \to \alpha$ in which E is the symbol on the left hand side. In fact, if such condition is not satisfied, the grammar can easily be transformed in polynomial time into one that generates the same set of marked-up documents, and in which the condition holds.

The set $docs(\mathbf{P}, S)$ of *marked-up documents* generated by $\mathbf{P}$ starting from a symbol S is inductively defined as follows:

- If S is a terminal, then $docs(\mathbf{P}, S) = S$.
- If S is an element and $(S \to \alpha) \in \mathbf{P}$, then

$$\begin{aligned} docs(\mathbf{P}, S) = \{ &\texttt{<}S\texttt{>}\, d_1 \cdots d_k \,\texttt{</}S\texttt{>} \mid \\ &\exists \sigma \in \alpha \text{ such that } \sigma = S_1 \cdots S_k \text{ and} \\ &d_i \in docs(\mathbf{P}, S_i), \text{ for } i \in \{1, \ldots, k\}\} \end{aligned}$$

The set of marked-up documents generated by a DTD $\mathcal{D} = (\mathbf{E}, \mathbf{T}, \mathbf{P}, I)$ is given by $docs(\mathbf{P}, I)$.

2.2 Equivalences and Inclusions between DTDs

Given two DTDs, a fundamental problem is to determine whether they are equivalent in some sense, i.e. whether they define the same sets of documents [Wood, 1995; Raymond *et al.*, 1995]. Here, we consider a more general problem, which is that of checking various forms of language inclusion (instead of equivalence). The most basic form of inclusion (equivalence) is inclusion (equality) of the sets of marked-up documents generated by the two DTSs. In general, when comparing DTDs we assume without loss of generality that they are over the same alphabets of terminals and elements.

Formally, let $\mathcal{D}_1 = (\mathbf{E}, \mathbf{T}, \mathbf{P}_1, I_1)$ and $\mathcal{D}_2 = (\mathbf{E}, \mathbf{T}, \mathbf{P}_2, I_2)$ be two DTDs. We say that $\mathcal{D}_1$ is *strongly included* in $\mathcal{D}_2$, denoted with $\mathcal{D}_1 \preceq_s \mathcal{D}_2$, if $docs(\mathbf{P}_1, I_1) \subseteq docs(\mathbf{P}_2, I_2)$.

For determining strong inclusion, the names of the start and end tags that constitute the markup of the two documents play a fundamental role.

In some cases, however, the actual names of the tags may not be relevant while the document structure imposed by the tags is of importance. The form of inclusion obtained by ignoring the names of tags and considering only their positions is called *structural inclusion* [Wood, 1995]. One DTDs is structurally included into another if, when we replace in every document generated by the DTDs all start and end tags with the unnamed tags `<>` and `</>` respectively, the resulting sets for the two DTDs are one included into the other.

While the restrictions imposed by strong inclusion may be too strict in some cases, structural inclusion, which ignores completely all tag names, may be too weak. A natural generalization of these two concepts is obtained by considering a spectrum of possible inclusions, of which strong and structural inclusion are just the two extremes. The different forms of inclusion are obtained by considering certain tag names as equal, and others as different, when confronting documents. This

```
<!DOCTYPE Mail [
<!ELEMENT Mail          (From, To, Subject, Body)>
<!ELEMENT From          (Address)>
<!ELEMENT To            (Address)+>
<!ELEMENT Subject       (#PCDATA)>
<!ELEMENT Body          (#PCDATA)>
<!ELEMENT Address       (#PCDATA)> ]>
```

Figure 1: DTD M for mail documents

allows us to parameterize inclusion (and therefore equivalence) of DTDs with respect to an equivalence relation on the set of tag names.

Formally, we consider an equivalence relation $\mathcal{R}$ on the set $\mathbf{E}$ of nonterminal symbols. For an element $E \in \mathbf{E}$, we denote by $[E]_\mathcal{R}$ the equivalence class of E with respect to $\mathcal{R}$. Given a DTD $\mathcal{D} = (\mathbf{E}, \mathbf{T}, \mathbf{P}, I)$ and such an equivalence relation $\mathcal{R}$, we inductively define the set $docs_\mathcal{R}(\mathbf{P}, S)$ of *$\mathcal{R}$-marked-up documents* generated by $\mathbf{P}$ starting from a symbol S as follows:

- If S is a terminal, then $docs_\mathcal{R}(\mathbf{P}, S) = S$.
- If S is an element and $(S \to \alpha) \in \mathbf{P}$, then

$$\begin{array}{l} docs_\mathcal{R}(\mathbf{P}, S) \; = \\ \quad \{\texttt{<}[S]_\mathcal{R}\texttt{>}\, d_1 \cdots d_k \,\texttt{</}[S]_\mathcal{R}\texttt{>} \mid \\ \quad\; \exists \sigma \in \alpha \text{ such that } \sigma = S_1 \cdots S_k \text{ and} \\ \quad\; d_i \in docs_\mathcal{R}(\mathbf{P}, S_i), \text{ for } i \in \{1, \ldots, k\}\} \end{array}$$

The set of $\mathcal{R}$-marked-up documents generated by a DTD $\mathcal{D} = (\mathbf{E}, \mathbf{T}, \mathbf{P}, I)$ is given by $docs_\mathcal{R}(\mathbf{P}, I)$.

For two DTDs $\mathcal{D}_1 = (\mathbf{E}, \mathbf{T}, \mathbf{P}_1, I_1)$ and $\mathcal{D}_2 = (\mathbf{E}, \mathbf{T}, \mathbf{P}_2, I_2)$ and an equivalence relation $\mathcal{R}$ on $\mathbf{E}$, we say that $\mathcal{D}_1$ is *$\mathcal{R}$-included* in $\mathcal{D}_2$, denoted with $\mathcal{D}_1 \preceq_\mathcal{R} \mathcal{D}_2$, if $docs_\mathcal{R}(\mathbf{P}_1, I_1) \subseteq docs_\mathcal{R}(\mathbf{P}_2, I_2)$.

Observe that, if we choose for $\mathcal{R}$ the equivalence relation in which all equivalence classes are singletons, we obtain strong inclusion. On the other hand, if $\mathcal{R}$ contains a single equivalence class constituted by the whole set $\mathbf{E}$, we obtain structural inclusion.

3 Representing DTDs and Reasoning on them

Let us introduce the logic $\mathcal{DL}$ which we use for formalizing DTDs and which is a simplified version of the formalism in [Calvanese *et al.*, 1995]. The syntax of $\mathcal{DL}$ is as follows:

$$\begin{array}{lcl} C & \longrightarrow & A \mid \top \mid \bot \mid \neg C \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \\ & & \forall R.C \mid \exists R.C \mid \exists^{\leq n} Q.C \mid \exists^{\leq n} Q^-.C \mid \\ & & (Q_1 \subseteq Q_2) \mid (Q_1^- \subseteq Q_2^-) \mid wf(R) \\ Q & \longrightarrow & P \mid Q_1 \cup Q_2 \mid Q_1 \cap Q_2 \mid Q_1 \setminus Q_2 \\ R & \longrightarrow & Q \mid R_1 \cup R_2 \mid R_1 \circ R_2 \mid R^- \mid R^* \mid id(C) \end{array}$$

where we denote concept names by A, arbitrary concepts by C, role names by P, *basic roles* (i.e. roles obtained by union, intersection and difference of role names) by Q, and arbitrary roles by R, all possibly with subscripts. The semantics of the constructs above is the standard one, except for the construct $wf(R)$, called *well-founded*, which is interpreted as those objects that are the initial point of only finite R-chains. Formally

$$\begin{array}{r} (wf(R))^\mathcal{I} = \{o_0 \in \Delta^\mathcal{I} \mid \forall o_1, o_2, \ldots \text{(ad infinitum)} \\ \exists o_i : \; (o_i, o_{i+1}) \notin R^\mathcal{I}\}. \end{array}$$

A *$\mathcal{DL}$ knowledge base* is a set of assertions of the form

$$C_1 \;\; \sqsubseteq \;\; C_2,$$

where C_1 and C_2 are arbitrary concepts without any restrictions. We use also $C_1 \equiv C_2$ as an abbreviation for the pair of assertions $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_1$.

An interpretation $\mathcal{I}$ *satisfies* the assertion $C_1 \sqsubseteq C_2$ if $C_1^\mathcal{I} \subseteq C_2^\mathcal{I}$. An interpretation is a *model* of a knowledge base $\mathcal{K}$ if it satisfies all assertions in $\mathcal{K}$[1]. Typical reasoning services (i.e. subsumption, satisfiability, logical implication) in $\mathcal{DL}$ are EXPTIME-complete [Calvanese *et al.*, 1995; Calvanese, 1996].

Let $\mathbf{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_k\}$ be a finite collection of DTDs. We assume without loss of generality that all DTDs in the collection share the same alphabets $\mathbf{T}$ of terminals and $\mathbf{E}$ of elements, i.e. that $\mathcal{D}_i = (\mathbf{E}, \mathbf{T}, \mathbf{P}_i, I_i)$, for $i \in \{1, \ldots, k\}$. We describe now how to construct from $\mathbf{D}$ a $\mathcal{DL}$ knowledge base $\mathcal{K}$ capable of fully capturing the various structural aspects of the DTDs in $\mathbf{D}$.

Independently from the particular collection of DTDs, $\mathcal{K}$ contains special assertions that model general structural properties of marked-up documents. Specifically, $\mathcal{K}$ contains the following assertions:

$$\begin{array}{rcl} \texttt{DStruc} & \equiv & \forall(\texttt{f} \cup \texttt{r}).\texttt{DStruc} \sqcap \\ & & \exists^{\leq 1}\texttt{f}.\top \sqcap \exists^{\leq 1}\texttt{r}.\top \sqcap \exists^{\leq 1}(\texttt{f} \cup \texttt{r})^-.\top \;\sqcap \\ & & wf(\texttt{f} \cup \texttt{r}) \\ \texttt{Tag} & \sqsubseteq & \texttt{DStruc} \sqcap \forall(\texttt{f} \cup \texttt{r}).\bot \\ \texttt{Terminal} & \sqsubseteq & \texttt{DStruc} \sqcap \forall(\texttt{f} \cup \texttt{r}).\bot \sqcap \neg\texttt{Tag} \end{array}$$

The concept `DStruc` introduces two distinguished roles `f` and `r` (standing for "first" and "rest" respectively) which are used to represent the tree-like form of marked-up documents by exploiting the standard encoding of n-ary trees as binary trees. The assertion on `DStruc` imposes functionality of `f` and `r` and the existence of at most one predecessor, hence enforcing a binary tree structure on

[1]This means that we adopt *descriptive semantics* for cycles.

$$
\begin{array}{rcl}
\mathtt{Mail}_M & \equiv & \mathtt{DStruc} \sqcap \exists \mathtt{f}.\mathtt{StartMail} \sqcap \exists(\mathtt{r} \circ id(\exists \mathtt{f}.\mathtt{From}_M) \circ \mathtt{r} \circ id(\exists \mathtt{f}.\mathtt{To}_M) \circ \mathtt{r} \\
& & \circ id(\exists \mathtt{f}.\mathtt{Subject}_M) \circ \mathtt{r} \circ id(\exists \mathtt{f}.\mathtt{Body}_M) \circ \mathtt{r}).\mathtt{EndMail} \\
\mathtt{From}_M & \equiv & \mathtt{DStruc} \sqcap \exists \mathtt{f}.\mathtt{StartFrom} \sqcap \exists(\mathtt{r} \circ id(\exists \mathtt{f}.\mathtt{Address}_M) \circ \mathtt{r}).\mathtt{EndFrom} \\
\mathtt{To}_M & \equiv & \mathtt{DStruc} \sqcap \exists \mathtt{f}.\mathtt{StartTo} \sqcap \exists(\mathtt{r} \circ id(\exists \mathtt{f}.\mathtt{Address}_M) \circ \mathtt{r} \circ (id(\exists \mathtt{f}.\mathtt{Address}_M) \circ \mathtt{r})^*).\mathtt{EndTo} \\
\mathtt{Subject}_M & \equiv & \mathtt{DStruc} \sqcap \exists \mathtt{f}.\mathtt{StartSubject} \sqcap \exists(\mathtt{r} \circ id(\exists \mathtt{f}.\mathtt{\#PCDATA}) \circ \mathtt{r}).\mathtt{EndSubject} \\
\mathtt{Body}_M & \equiv & \mathtt{DStruc} \sqcap \exists \mathtt{f}.\mathtt{StartBody} \sqcap \exists(\mathtt{r} \circ id(\exists \mathtt{f}.\mathtt{\#PCDATA}) \circ \mathtt{r}).\mathtt{EndBody} \\
\mathtt{Address}_M & \equiv & \mathtt{DStruc} \sqcap \exists \mathtt{f}.\mathtt{StartAddress} \sqcap \exists(\mathtt{r} \circ id(\exists \mathtt{f}.\mathtt{\#PCDATA}) \circ \mathtt{r}).\mathtt{EndAddress} \\
\mathtt{\#PCDATA} & \sqsubseteq & \mathtt{Terminal}
\end{array}
$$

Figure 2: Knowledge base derived from the DTD M

the $(\mathtt{f} \cup \mathtt{r})^*$-connected components of models of the knowledge base. Observe that the use of the well-foundedness construct is essential to impose finiteness and acyclicity of such connected components.

The symbols that appear in the DTDs and the element type definitions are then encoded in $\mathcal{K}$ as follows:

- For each terminal $F \in \mathbf{T}$, $\mathcal{K}$ contains an assertion

$$F \ \sqsubseteq \ \mathtt{Terminal}.$$

- For each element $E \in \mathbf{E}$, $\mathcal{K}$ contains two assertions

$$
\begin{array}{rcl}
\mathtt{Start}E & \sqsubseteq & \mathtt{Tag} \\
\mathtt{End}E & \sqsubseteq & \mathtt{Tag},
\end{array}
$$

  where $\mathtt{Start}E$ and $\mathtt{End}E$ represent start and end tags.

- For each $\mathcal{D}_i \in \mathbf{D}$, and for each element E, such that $(E \rightarrow \alpha) \in \mathbf{P}_i$[2], $\mathcal{K}$ contains the assertion:

$$E_{\mathcal{D}_i} \ \equiv \ \mathtt{DStruc} \sqcap \exists \mathtt{f}.\mathtt{Start}E \sqcap \exists(\mathtt{r} \circ \tau(\alpha)).\mathtt{End}E$$

  with $\tau(\alpha)$ defined inductively as:

$$
\begin{array}{rcl}
\tau(\varepsilon) & = & id(\top) \\
\tau(S) & = & id(\exists \mathtt{f}.cn(\mathcal{D}_i, S)) \circ \mathtt{r} \\
\tau(\alpha_1|\alpha_2) & = & \tau(\alpha_1) \cup \tau(\alpha_2) \\
\tau(\alpha_1, \alpha_2) & = & \tau(\alpha_1) \circ \tau(\alpha_2) \\
\tau(\alpha^*) & = & \tau(\alpha)^*
\end{array}
$$

  where $cn(\cdot,\cdot)$ is a mapping that associates to each pair constituted by a DTD $\mathcal{D}_i$ and a symbol S a concept name as follows:

$$cn(\mathcal{D}_i, S) = \begin{cases} E_{\mathcal{D}_i} & \text{if } S = E \text{ for an element } E \in \mathbf{E} \\ F & \text{if } S = F \text{ for a terminal } F \in \mathbf{T} \end{cases}$$

The role $\tau(\alpha)$ reflects the structure imposed by α on the parts of a document that are defined by $E \rightarrow \alpha$. It can be explained in terms of an encoding of the tree representing the marked-up document into a binary tree.

Observe that while for each tag in the collection of DTDs we have a unique concept name, the information about the DTD a given element belongs to is explicitly carried out in the knowledge base. Indeed, for each element E we have introduced two concept names $\mathtt{Start}E, \mathtt{End}E$ representing its tags, and one concept name $E_{\mathcal{D}_i}$ for each DTD $\mathcal{D}_i$ containing a definition of E.

We stress that for each $E_{\mathcal{D}_i}$ the well-foundedness construct in the assertion on $\mathtt{DStruc}$ ensures the following: Given a model $\mathcal{M}$ of $\mathcal{K}$, it is possible to determine whether an object o is an instance of $E_{\mathcal{D}_i}$ by taking into account only the structure of the $(\mathtt{f} \cup \mathtt{r})^*$ connected component of $\mathcal{M}$ containing o, and the concepts representing tags and terminals on such component. This property is essential in order to obtain the desired correspondence between reasoning on the DTDs in $\mathbf{D}$ and reasoning on $\mathcal{K}$.

Figure 2 shows the knowledge base corresponding to the DTD M described in Figure 1. We have omitted the assertions that are independent from the particular DTD, introduced above, as well as those for the concepts that represent the various start and end tags.

The knowledge base $\mathcal{K}$ can directly be used to determine strong inclusion between DTDs.

Theorem 1 *Let $\mathcal{D}_i$ and $\mathcal{D}_j$ be two DTDs in $\boldsymbol{D}$, and $\mathcal{K}$ the $\mathcal{DL}$ knowledge base derived from $\boldsymbol{D}$ as specified above. Then $\mathcal{D}_i$ is strongly included in $\mathcal{D}_j$ if and only if $cn(\mathcal{D}_i, I_i)$ is subsumed by $cn(\mathcal{D}_j, I_j)$ in $\mathcal{K}$.*

The knowledge base $\mathcal{K}$ can also be extended in order to verify the other forms of inclusions introduced in Section 2. Let $\mathcal{R} = \{\{E_1^1, \ldots, E_{n_1}^1\}, \ldots, \{E_1^m, \ldots, E_{n_m}^m\}\}$ be an equivalence relation on the set $\mathbf{E}$ of elements. We obtain the knowledge base $\mathcal{K}_\mathcal{R}$ from $\mathcal{K}$ by adding for each equivalence class $\{E_1^j, \ldots, E_{n_j}^j\}$ and for each element E_i^j, with $i \in \{1, \ldots, n_j - 1\}$, the assertions:

$$
\begin{array}{rcl}
\mathtt{Start}E_i^j & \equiv & \mathtt{Start}E_{i+1}^j \\
\mathtt{End}E_i^j & \equiv & \mathtt{End}E_{i+1}^j
\end{array}
$$

With these assertions we are essentially imposing the equivalence of all the concepts representing tags of elements belonging to each set $\{E_1^j, \ldots, E_{n_j}^j\}$. Therefore, when reasoning on $\mathcal{K}_\mathcal{R}$ the differences between the various tags associated to equivalent elements are ignored, coherently with the notion of $\mathcal{R}$-inclusion.

Theorem 2 *Let $\mathcal{D}_i$ and $\mathcal{D}_j$ be two DTDs in $\boldsymbol{D}$, $\mathcal{R}$ an equivalence relation on $\boldsymbol{E}$, and $\mathcal{K}_\mathcal{R}$ the $\mathcal{DL}$ knowledge*

[2]We assume without loss of generality that every element appearing in $\mathbf{P}_i$, appears also as the left hand side of some element type definition in $\mathbf{P}_i$.

base derived from $\boldsymbol{D}$ *as specified above. Then* $\mathcal{D}_i$ *is* $\mathcal{R}$*-included in* D_j *if and only if* $cn(\mathcal{D}_i, I_i)$ *is subsumed by* $cn(\mathcal{D}_j, I_j)$ *in* $\mathcal{K}_{\mathcal{R}}$.

From decidability in deterministic exponential time of logical implication in $\mathcal{DL}$ [Calvanese *et al.*, 1995] we obtain as an immediate consequence an EXPTIME upper bound for $\mathcal{R}$-inclusion and $\mathcal{R}$-equivalence between DTDs.

Corollary 3 *$\mathcal{R}$-inclusion and $\mathcal{R}$-equivalence between two DTDs can be verified in deterministic exponential time in the size of the DTDs.*

As mentioned, the previously known algorithms for structural equivalence were doubly exponential, while by the theorem above we obtain a single exponential upper bound.

4 Conclusions

Several recent papers dealing with the problem of querying the World Wide Web argue that the current techniques for representing and reasoning on document structures should be improved. We have provided a view of DTDs as concepts of the expressive Description Logic $\mathcal{DL}$, and we have demonstrated that this approach is indeed very effective for both faithfully representing document structures, and answering several open questions regarding DTD equivalence checking. By exploiting the constructs of $\mathcal{DL}$, we are able to integrate into the structure of documents also aspects related to the semantics of the information contained in them. For example, the so called *attribute lists* of DTD elements can be modeled easily in $\mathcal{DL}$. As another example, if part of a document (corresponding to a terminal symbol T in the DTD) includes a table with information about, say, departments and employees, this can be represented by adding suitable properties to the concept corresponding to T. We can also represent links to other documents, such as those typically found in the Web, by means of a special concept with suitable roles for the name of the link and the associated anchor. Obviously, by means of suitable assertions we can constrain the anchor to point to a document of a specific DTD.

We are extending our work along two directions. On one hand, we aim at capturing more aspects of DTDs in order to represent, for example, other properties of documents (attribute list in the terminology of SGML), exceptions (as described in [Wood, 1995]), or constraints on the number of occurrences of a certain pattern in an element definition. On the other hand, the deductive power of $\mathcal{DL}$ allows us to study new types of reasoning on DTDs, such as further forms of parameterized equivalence (e.g. abstracting from the definition of a specified element), or document classification (infer which is the DTD that best matches a given marked document among a set of candidates).

References

[Calvanese *et al.*, 1995] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Structured objects: Modeling and reasoning. In *Proc. of the 4th Int. Conf. on Deductive and Object-Oriented Databases (DOOD-95)*, number 1013 in Lecture Notes in Computer Science, pages 229–246. Springer-Verlag, 1995.

[Calvanese, 1996] Diego Calvanese. Finite model reasoning in description logics. In Luigia C. Aiello, John Doyle, and Stuart C. Shapiro, editors, *Proc. of the 5th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-96)*. Morgan Kaufmann, Los Altos, 1996.

[Christophides *et al.*, 1994] V. Christophides, S. Abiteboul, S. Cluet, and M. Scholl. From structured documents to novel query facilities. In R. T. Snodgrass and M. Winslett, editors, *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, pages 313–324, Minneapolis (Minnesota, USA), 1994.

[Int, 1986] International Organization for Standardization. *ISO-8879: Information processing – Text and office systems – Standard Generalized Markup Language (SGML)*, October 1986.

[Konopnicki and Shmueli, 1995] D. Konopnicki and O. Shmueli. W3QS: A query system for the World Wide Web. In *Proc. of the 21th Int. Conf. on Very Large Data Bases (VLDB-95)*, pages 54–65, 1995.

[Lakshmanan *et al.*, 1996] L. Lakshmanan, F. Sadri, and I. N. Subramanian. A declarative language for querying and restructuring the Web. In *Proc. of the 6th Int. Workshop on Reasearch Issues in Data Enginnering: Interoperability of Nontraditional Database Systems*. IEEE Computer Science Press, 1996.

[Mendelzon *et al.*, 1996] A. Mendelzon, G. A. Mihaila, and T. Milo. Querying the World Wide Web. `ftp://db.toronto.edu/pub/papers/websql.ps`, 1996.

[Quass *et al.*, 1995] D. Quass, A. Rajaraman, I. Sagiv, J. Ullman, and J. Widom. Querying semistructured heterogeneous information. In *Proc. of the 4th Int. Conf. on Deductive and Object-Oriented Databases (DOOD-95)*, pages 319–344. Springer-Verlag, 1995.

[Raymond *et al.*, 1995] D. R. Raymond, F.W. Tompa, and D. Wood. From data implementation to data model: Meta-semantic issues in the evolution of SGML. *Computer Standards and Interfaces*, 1995.

[Wood, 1995] Derick Wood. Standard Generalized Markup Language: Mathematical and philosophical issues. In Jan van Leeuwen, editor, *Computer Science Today, Recent Trends and Developments*, number 1000 in Lecture Notes in Computer Science, pages 344–365. Springer-Verlag, 1995.