Representing and Reasoning on SGML Documents

Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini

Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italy
{calvanese,degiacomo,lenzerini}@dis.uniroma1.it

Abstract. In this paper, we address the issue of representing and reasoning about documents for which an explicit structure is provided. Specifically, we devise a framework where Document Type Definitions (DTDs) expressed in the Standard Generalized Markup Language (SGML) are formalized in an expressive Description Logic equipped with sound, complete, and terminating inference procedures. In this way, we provide a general reasoning mechanism that enables various reasoning tasks on DTDs, including the verification of typical forms of equivalences between DTDs, such as strong equivalence and structural equivalence, as well as parametric versions of these equivalences. Notably, this general reasoning mechanism allows for verifying structural equivalence in worst case deterministic exponential time, in contrast to the known algorithms which are double exponential. As a whole, the study in this paper provides some of the fundamental building blocks for developing articulated inference systems that support tasks involving the intelligent navigation of large document databases such as the World Wide Web.

1 Introduction

In this paper, we address the issue of representing and reasoning on documents for which an explicit structure is provided. The possibility of representing and reasoning on the document structure is advocated by research in both Knowledge Representation and Databases. In particular, the view of the World Wide Web as a large information system constituted by a collection of interconnected documents, and the increasing popularity of private intranets among large companies or institutions for keeping documentation online, is stimulating much work on information retrieval from large document databases (see for example [9,13,10,11]).

This work points out that being able to represent and reason on the structure of documents placed in document databases helps in several tasks related to information retrieval. For example, it enables to improve both the precision of the information retrieved by providing flexible additional selection criteria, and the efficiency of the retrieval process, by allowing for retrieving just a short description of a large document to decide its relevance, instead of the document itself [8,12,1,6,14,13].

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

Fig. 1. DTD M for mail documents

The structure of a document is typically made explicit by using special tags to mark its various parts. One of the most prominent formalisms for defining marked-up documents is the *Standard Generalized Markup Language* (SGML) [7]. In SGML, the structure of marked-up documents is described by means of *Document Type Definitions* (DTDs) which assert the set of "rules" that each document of a given document type must conform to. SGML DTDs have been used to define wide range of document types, from very general ones, such as generic HTML documents, to very specific ones, e.g. a specific form of email messages.

In this paper, we show a formalization of SGML DTDs in terms of an expressive Description Logic, $\mathcal{DL}$, equipped with sound, complete, and terminating inference procedures. This logic includes non-first-order constructs, such as reflexive-transitive closure and well-foundedness, which play a crucial role in the formalization. The inference procedures for such logic provide us with a general reasoning mechanism that enables effective reasoning tasks on DTDs. These include the verification of typical forms of equivalences between DTDs [17,15], such as: strong equivalence, i.e. whether two DTDs define the same sets of marked-up documents; structural equivalence, a weaker form of equivalence abstracting from tag names in the documents; and parametric versions of these equivalences. Notably, this general reasoning mechanism allows for verifying structural equivalence in worst case deterministic exponential time, in contrast to the known algorithms which are double exponential.

The paper is organized as follows. In Section 2, SGML DTDs and documents are introduced. In Section 3, the basic reasoning tasks on DTDs are defined. In Section 4, the Description Logic $\mathcal{DL}$ is presented and the formalization of DTDs (and related reasoning tasks) within $\mathcal{DL}$ is developed. Finally in Section 5 some conclusions are drawn.

2 SGML DTDs and Documents

An SGML document consists of an SGML prologue and a marked-up document. 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. The logical components of a document are called *elements.*

The fundamental characteristics of DTDs can be formalized by means of Extended Context Free Grammars (ECFGs) [17]. Marked-up documents 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 *non-terminal 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. When no ambiguity may arise, we identify α with the set of words generated by the regular expression that α represents. Additionally, in content models, the following standard abbreviations are used:

$$\alpha_1 \& \alpha_2 = (\alpha_1, \alpha_2)|(\alpha_2, \alpha_1) \qquad \alpha? = \varepsilon|\alpha \qquad \alpha^+ = \alpha, \alpha^*.$$

We observe that while "?" and "+" pose no particular problem, expanding "&" may in the worst case lead to an exponential increase in the size of the DTD.

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 4, makes it easy 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 $docs(\mathbf{P}, S)$ is the set of sequences `<`S`>` $d_1 \cdots d_k$ `</`S`>`, where `<`S`>` and `</`S`>` are the start and end tags associated

to the element S, and $d_1, \ldots, d_k$ are documents generated by an instance of the regular expression α. Formally

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

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

3 Basic Reasoning Tasks on 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 [17,15]. 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 DTDs. 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[1]. 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* [17]. 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.

Structural equivalence of two DTDs is decidable, but the known algorithms take time doubly exponential in the size of the two DTDs [17]. This complexity bound holds if one does not consider the "&" operator, which, if expanded may lead to an additional exponential blowup.

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 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

[1] In general, when comparing DTDs we assume without loss of generality that they are over the same alphabets of terminals and elements.

Concepts C	**Syntax**	**Semantics**
concept name	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
top	$\top$	$\Delta^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
conjunction	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
disjunction	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
universal quantif.	$\forall R.C$	$\{o \mid \forall o' \colon (o,o') \in R^{\mathcal{I}} \rightarrow o' \in C^{\mathcal{I}}\}$
existential quantif.	$\exists R.C$	$\{o \mid \exists o' \colon (o,o') \in R^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}}\}$
qualified number	$\exists^{\leq n} Q.C$	$\{o \mid \sharp\{o' \mid (o,o') \in Q^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}}\} \leq n\}$
restrictions	$\exists^{\leq n} Q^-.C$	$\{o \mid \sharp\{o' \mid (o,o') \in (Q^-)^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}}\} \leq n\}$
well-founded	$wf(R)$	$\{o_0 \mid \forall o_1, o_2, \ldots \text{(ad infinitum)}\ \exists i \geq 0 \colon (o_i, o_{i+1}) \notin R^{\mathcal{I}}\}$
Roles R	**Syntax**	**Semantics**
role name	P	$P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
union	$R_1 \cup R_2$	$R_1^{\mathcal{I}} \cup R_2^{\mathcal{I}}$
concatenation	$R_1 \circ R_2$	$R_1^{\mathcal{I}} \circ R_2^{\mathcal{I}}$
inverse	R^-	$\{(o,o') \mid (o',o) \in R^{\mathcal{I}}\}$
refl. trans. closure	R^*	$(R^{\mathcal{I}})^*$
identity	$id(C)$	$\{(o,o) \mid o \in C^{\mathcal{I}}\}$

Table 1. Syntax and semantics of $\mathcal{DL}$ concept and role constructs.

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 $(S \rightarrow \alpha) \in \mathbf{P}$, then

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

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.

4 Description Logic for DTDs

Let us introduce the logic $\mathcal{DL}$ which we use for formalizing DTDs and which is a simplified version of the formalisms in [5,4]. The syntax and semantics of $\mathcal{DL}$ are shown in Table 1, where we denote concept names by A, arbitrary concepts by C, role names by P, and arbitrary roles by R, all possibly with subscripts. The semantics of the $\mathcal{DL}$ constructs 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.

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 $\mathcal{K}$[2]. Typical reasoning services (i.e. subsumption, satisfiability, logical implication) in $\mathcal{DL}$ are EXPTIME-complete [5,4].

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 $\mathbf{D}$. The knowledge base $\mathcal{K}$ is constituted by three parts, called $\mathcal{K}_0, \mathcal{K}_1$ and $\mathcal{K}_2$, respectively, which we now describe.

Independently from the particular collection of DTDs, $\mathcal{K}_0$ contains special assertions that model general structural properties of marked-up documents:

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

Every instance of `DStruc` represents an SGML document. Every instance of `Tag` represents a tag in $\mathbf{D}$, either a start or an end tag. Finally, every instance of `Terminal` represents a terminal symbol in $\mathbf{D}$.

The concept `DStruc` is defined in terms of the roles `f` and `r` (standing for "first" and "rest" respectively). The components of a document are found by following the $(\texttt{f} \cup \texttt{r})^{+}$-links. More precisely, the first component of a document d is its `f`-filler, the second component is its $(\texttt{r} \circ \texttt{f})$-filler, the third component is its $(\texttt{r} \circ \texttt{r} \circ \texttt{f})$-filler, and the last component is its $\texttt{r}^h$-filler, for some $h > 0$[3].

Observe that the definition of `DStruc` imposes that `f` and `r` are functions, and that every instance of `DStruc` has at most one $(\texttt{f} \cup \texttt{r})$-predecessor, hence enforcing a binary tree structure on the $(\texttt{f} \cup \texttt{r})^{*}$-connected components in the models of the knowledge base. Notably, the use of the well-foundedness construct is essential to impose finiteness and acyclicity of such connected components.

$\mathcal{K}_1$ is used to represent the specific tags and terminal symbols appearing in $\mathbf{D}$. In particular,

[2] This means that we adopt *descriptive semantics* for cycles.
[3] $\texttt{r}^h$ denotes $\texttt{r} \circ \cdots \circ \texttt{r}$ (h times).

– For each terminal $F \in \mathbf{T}$, $\mathcal{K}_1$ contains an assertion

$$F \quad \sqsubseteq \quad \texttt{Terminal}.$$

– For each element $E \in \mathbf{E}$, $\mathcal{K}_1$ contains two assertions

$$\texttt{Start}E \quad \sqsubseteq \quad \texttt{Tag} \qquad\qquad\qquad \texttt{End}E \quad \sqsubseteq \quad \texttt{Tag},$$

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

$\mathcal{K}_2$ encodes the knowledge about the various production rules in $\mathbf{D}$. In particular, for each $\mathcal{D}_i \in \mathbf{D}$, and for each element E, such that $(E \to \alpha) \in \mathbf{P}_i$, $\mathcal{K}_2$ contains the assertion:

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

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

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

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:

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

Note that the first component (the $\texttt{f}$-filler) of every instance of $E_{\mathcal{D}_i}$ is its start tag, whereas the last component (the $\texttt{r}^h$-filler) is its end tag. The remaining components (the $(\texttt{r}^k \circ \texttt{f})$-fillers, with $k < h$) are determined by the complex role $\tau(\alpha)$. Indeed $\tau(\alpha)$ reflects the structure imposed by α on the parts of a document that are defined by $E \to \alpha$, and can be explained in terms of an encoding of the tree representing the marked-up document into a binary tree.

Observe that for each element E we have introduced two concept names $\texttt{Start}E, \texttt{End}E$ representing its tags. Thus, for each tag in the collection $\mathbf{D}$ of DTDs there is a unique pair of corresponding concepts in $\mathcal{K}$. On the contrary, the information about the DTD a given element belongs to is explicitly carried out in the knowledge base. Indeed, there is one concept $E_{\mathcal{D}_i}$ in $\mathcal{K}$ for each DTD $\mathcal{D}_i$ in $\mathbf{D}$ containing the definition of an element E.

We stress that in each model of $\mathcal{K}$, the extension of every $E_{\mathcal{D}_i}$ is completely determined by the extension of the concepts representing tags and terminal symbols. In other words, 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 $(\texttt{f} \cup \texttt{r})^*$ connected component of $\mathcal{M}$ containing o. This property, which is ensured by the well-foundedness construct in the assertion on $\texttt{DStruc}$, is essential in order to obtain the desired correspondence between reasoning on the DTDs in $\mathbf{D}$ and reasoning on $\mathcal{K}$.

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

Fig. 2. The $\mathcal{K}_2$ part of the knowledge base $\mathcal{K}$ derived from the DTD M

Figure 2 shows the $\mathcal{K}_2$ part of the knowledge base $\mathcal{K}$ corresponding to the DTD M described in Figure 1. One can easily derive the $\mathcal{K}_0$ and $\mathcal{K}_1$ parts of $\mathcal{K}$.

The knowledge base $\mathcal{K}$ corresponding to a collection $\mathbf{D}$ of documents can directly be used to determine strong inclusion between DTDs belonging to $\mathbf{D}$.

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 $\mathcal{R} = \{\{E_1^1, \dots, E_{n_1}^1\}, \dots, \{E_1^m, \dots, E_{n_m}^m\}\}$ 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 $\{E_1^j, \dots, E_{n_j}^j\}$ and for each element E_i^j, with $i \in \{1, \dots, n_j-1\}$, the assertions:

$$\mathtt{Start}E_i^j \equiv \mathtt{Start}E_{i+1}^j \qquad\qquad \mathtt{End}E_i^j \equiv \mathtt{End}E_{i+1}^j$$

With these assertions we are essentially imposing the equivalence of all the concepts representing tags of elements belonging to each set $\{E_1^j, \dots, E_{n_j}^j\}$. 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 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 $\mathcal{DL}$ [5] we obtain as an immediate consequence an EXPTIME upper bound for $\mathcal{R}$-inclusion and $\mathcal{R}$-equivalence between DTDs. This results also in an exponential improvement over previously known algorithms for checking structural equivalence.

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.*

5 Conclusions

Several recent papers dealing with the problem of retrieving information from a document database such as 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 some 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, 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.

The framework presented in this paper for representing and reasoning on structured documents provides a notable example of handling objects composed of different parts. The *part-whole* relation is seen as having a special importance in several applications [6,3,12]. $\mathcal{DL}$, by means of the reflexive-transitive closure and the well-foundedness constructs, is able to capture fundamental aspects of the part-whole relation [12,2,16] as shown in [5].

Two further research directions are worth pursuing. On the one hand, further aspects of DTDs could be captured in order to represent, for example, other properties of documents, exceptions (as described in [17]), and 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 one 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) and document classification (infer which is the DTD that best matches a given marked document among a set of candidates).

References

1. J. Åberg. Creating a description logics knowledge base for world wide web documents. Technical Report LiTH-IDA-Ex-9641, Department of Computer and Information Science, Linköping University, Sweden, 1996.
2. A. Artale, E. Franconi, and N. Guarino. Open problems with part-whole relations. In *Proc. of the 1996 Description Logic Workshop (DL-96)*, number WS-96-05, pages 70–73. AAAI Press, 1996.
3. A. Artale, E. Franconi, N. Guarino, and L. Pazzi. Part-whole relations in object-centered systems: An overview. *Data and Knowledge Engineering*, 20:347–383, 1996.

4. D. Calvanese. Finite model reasoning in description logics. In *Proc. of the 5th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-96)*, pages 292–303. Morgan Kaufmann, 1996.
5. D. Calvanese, G. De Giacomo, and M. Lenzerini. Structured objects: Modeling and reasoning. In *Proc. of the 4th Int. Conf. on Deductive and Object-Oriented Databases (DOOD-95)*, number 1013 in LNCS, pages 229–246. Springer-Verlag, 1995.
6. 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, 1994.
7. International Organization for Standardization. *ISO-8879: Information processing – Text and office systems – Standard Generalized Markup Language (SGML)*, October 1986.
8. T. Kirk, A.Y. Levy, Y. Sagiv, and Divesh Srivastava. The Information Manifold. In *Proceedings of the AAAI 1995 Spring Symp. on Information Gathering from Heterogeneous, Distributed Enviroments*, pages 85–91, 1995.
9. Craig Knoblock and Alon Y. Levy, editors. *Proceedings of the AAAI 1995 Spring Symp. on Information Gathering from Heterogeneous, Distributed Enviroments*, number SS-95-08, Menlo Park (U.S.A.), 1995. AAAI Press/The MIT Press.
10. 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.
11. 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.
12. P. Lambrix. *Part-Whole Reasoning in Description Logic.* PhD thesis, Dep. of Computer and Information Science, Linköping University, Sweden, 1996.
13. A. Mendelzon, G. A. Mihaila, and T. Milo. Querying the World Wide Web. *International Journal on Digital Libraries*, 1(1):54–67, 1997.
14. 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.
15. 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.
16. U. Sattler. A concept language for an engineering application with part-whole relations. In A. Borgida, M. Lenzerini, D. Nardi, and B. Nebel, editors, *Working Notes of the 1995 Description Logics Workshop*, Technical Report, RAP 07.95, Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", pages 119–123, Rome (Italy), 1995.
17. D. Wood. Standard Generalized Markup Language: Mathematical and philosophical issues. In Jan van Leeuwen, editor, *Computer Science Today, Recent Trends and Developments*, number 1000 in LNCS, pages 344–365. Springer-Verlag, 1995.