Approximation from a Description Logic with disjunction to another without disjunction

Chan Le Duc, Nhan Le Thanh
Laboratoire I3S, Université de Nice-Sophia Antipolis
Email: cleduc@i3s.unice.fr, Nhan.Le-Thanh@unice.fr

Abstract

Even despite the studies carried out concerning Description Logics with conjunction, the structural approach for standard and non-standard inferences in Description Logics with disjunction remains a subject relatively unexplored. The first studies of approximation inference were made by S. Brandt, R.Küsters and A.Turhan. The authors have proposed a *double exponential* algorithm for computing the upper approximation from an $\mathcal{ALC}$-concept description to an $\mathcal{ALE}$-concept description ($\mathcal{ALC}$-$\mathcal{ALE}$). In many application areas, this double exponential algorithm is largely unsatisfactory. Firstly, this paper aims to study subsumption in $\mathcal{ALU}$ ($\mathcal{ELU}$) language. An approximation algorithm from an $\mathcal{ALU}$-concept description to $\mathcal{FL}_\neg$-concept description can be deduced directly from this subsumption. Secondly, this paper proposes an *exponential* algorithm for computing the upper approximation $\mathcal{ALC}$-$\mathcal{ALE}$. The technique used for this algorithm is based on the possibility of distributing the approximation computation over disjuncts and conjuncts.

1 Introduction

In electronic commerce, a transaction consists of document interchanges between partners [11]. Composition and interpretation of these documents require a vocabulary for each partner as well as a mechanism capable of translating one vocabulary into another. However, electronic commerce is a heterogeneous domain and consists of many different subdomains. Each subdomain uses its own vocabulary for internal document interchanges. At present, document interchange between subdomains in semantics tranparency poses a constant challenge. Fortunately, however, recent use of ontologies for design of such vocabularies can be considered as a solution for the problem. Each subdomain has its own vocabulary, all of which derive from a common ontology. These derived ontologies are probably designed with the use of different DL languages according to required expressiveness. In such a system, one of the essential questions is how to interprete a non-defined term found in a received document.

This study addresses the problem which arises when automatically translating from a derived ontology in an expressive DL into another ontology (sharing the same base concepts) in less expressive DL. The interpreting mechanism between derived ontologies, therefore, requires an efficient algorithm which would allow us to exactely or approximately compute a L_s-concept description (source) into L_d-concept description (destination). In the case where the precise interpretation does not exist, the approximative computation would allow us to propose a "nearest" L_d-concept description w.r.t subsumption relation.

We begin by investigating the subsumption problem of $\mathcal{ALU}$ in Section 3. In attempting to extend the automata theory method mentionned in [1] for $\mathcal{ALU}$ but obtained result was not as satisfactory as first hope: we could not represent an $\mathcal{ALU}$-concept description as a automaton. The result archived was instead a pushdown automaton. This allows us, however, to interprete the semantics of an $\mathcal{ALU}$-concept description as being tree-like in structure. This section is then terminated by an algorithm for computing approximation $\mathcal{ALU}$-$\mathcal{FL}_{\neg}$. This approximation is based on subsumption characterization in $\mathcal{ALU}$.

In section 4, we show a interesting property of *lcs* (least common subsumer) in $\mathcal{ALE}$ which states that for *propagated* $\mathcal{ALE}$ concept descriptions, the common conjuncts are also conjuncts of *lcs*. Nevertheless, before computing of $\mathcal{ALC}$-$\mathcal{ALE}$ approximation, necessary transformations must be carried out on $\mathcal{ALC}$-concept description. These transformations will allow us to distribute the approximation computation in $\mathcal{ALE}$ of an $\mathcal{ALC}$-concept description over its disjuncts and conjuncts. This technique is one of crucial points that will avoid the double exponential complexity in the approximation algorithm.

2 Description Logics

Concept descriptions are built from concept constructors, a set N_C of concept names and a set N_R of role names. The semantics of a concept description in some language is defined with the aide of the following interpretation $I=(\Delta, \cdot^I)$. The following table gives a summary of languages that are used in

this paper. Note that the largest language used in this paper is $\mathcal{ALC}$. This language allows concept descriptions to be negated whereas in $\mathcal{ALE}$ negation is only allowed in front of concept names.
Subsumption inference used in this paper is defined as follows : the concept description C is subsumed by the description D, written $C \sqsubseteq D$, iff $C^I \subseteq D^I$ holds for all interpretation $\mathcal{I}$.

Syntax	Semantics	$\mathcal{FL}_\neg$	$\mathcal{ALU}$	$\mathcal{ELU}$	$\mathcal{ALE}$	$\mathcal{ALC}$
$\top$	Δ	x	x	x	x	x
$\bot$	$\emptyset$	x	x	x	x	x
$C \sqcap D$	$\mathrm{C}^I \cap D^I$	x	x	x	x	x
$\forall r.C$	$\{x \in \Delta \vert \forall y{:}(x,y) \in r^I \rightarrow y \in C^I\}$	x	x		x	x
$\exists r.C$	$\{x \in \Delta \vert \exists y{:}(x,y) \in r^I \wedge y \in C^I\}$			x	x	x
$\neg A, A \in N_C$	$\Delta \setminus A^I$	x	x	x	x	x
$C \sqcup D$	$\mathrm{C}^I \cup D^I$		x	x		x
$\neg C$	$\Delta \setminus C^I$					x

3 Characterization of subsumption in $\mathcal{ALU}(\mathcal{ELU})$ and $\mathcal{ALU}$-$\mathcal{FL}_\neg$ approximation

We will begin by investigating the problem of subsumption in $\mathcal{ALU}(\mathcal{ELU})$. Firstly, let us provide the following definition in order to clarify the notion of approximation.

Definition 3.1. *Let L_s and L_d be two DLs, and let C be a L_s-concept description and D be a L_d– concept description. Then, D is called upper (lower) L_d-approximation of C (min(C), max(C) for short) if*
i) $C \sqsubseteq D$ ($D \sqsubseteq C$) and
ii) $C \sqsubseteq D'$ ($D' \sqsubseteq C$), $D' \sqsubseteq D$ ($D \sqsubseteq D'$) implies $D' \equiv D$ for all L_d-concept descriptions D'

Remarks: Let L_s=$\mathcal{ALC}$, L_d=$\mathcal{ALE}$. If there exists $min(C)$ then it is unique. Indeed, if D_1= $min(C)$ and D_2= $min(C)$ then $C \sqsubseteq D_1 \sqcap D_2 \sqsubseteq D_1$ and $C \sqsubseteq D_1 \sqcap D_2 \sqsubseteq D_2$. It means that $D_1 \equiv D_2$. But there may be many $max(C)$. For example, $C = \exists r.P \sqcup \forall r.Q$, $max'(C) = \exists r.P$, $max''(C) = \forall r.Q$.

Now, we will provide the following normalization definition allowing us to postpone processing conjunctions.

Definition 3.2. *A term is called $\forall$-normal iff it is disjunction of terms* *of the form :* $\forall r.(A \sqcup B)$ *where A,B are also* $\forall$*-normal terms and none of the following rules can be applied at any position in the term :*

$$\begin{array}{rcl} P \sqcup \neg P & \rightarrow & \top \\ E \sqcup \bot & \rightarrow & E \\ \forall r.\top & \rightarrow & \top \\ E \sqcup \top & \rightarrow & \top \end{array}$$

Remarks: i) $\forall$-*normal* terms do not contain conjunction.
ii) Any $\mathcal{ALU}$-concept description C can be represented as conjunctions of $\forall$-*normal* terms. Indeed this transformation can be made providing that two following properties : $(A \sqcap B) \sqcup C$=$(A \sqcup C) \sqcap (B \sqcup C)$ and $\forall w.(A \sqcap B) = \forall w.A \sqcap \forall w.B$. This transformation can take an exponential time in the size of C.
iii) Replacing $\forall$ and $\sqcup$ with respectively $\exists$ and $\sqcap$ in the definition, we will obtain the definition of $\exists$-*normal* term. In the same way, any $\mathcal{ELU}$-concept description can be represented as disjunctions of $\exists$-*normal* terms.
iv) We denote $Q_i \in \{P_i | P_i \in N_C\} \cup \{\neg P_i | P_i \in N_C\}, r_i \in N_R$.

Definition 3.3. *A is a* $\forall$*-normal term. The language of A, called L(A), is defined recursively as follows:*

$$\begin{array}{ll} (I) & L(Q) = \{(Q).\varepsilon\} = \{(Q)\}; L(Q_1 \sqcup Q_2) = \{(Q_1, Q_2).\varepsilon\} = \{(Q_1, Q_2)\} \\ (II) & L(A_1 \sqcup A_2) = \{L(A_1) \cup L(A_2)\} \\ (III) & L(\forall r_1.(A_1 \sqcup A_2))) = \{(L(A_1) \cup L(A_2)).r_1\} \end{array}$$

Remarks: i) $L(A)$ is equivalent to a pushdown automaton. ii) For all words $w \in L(A)$, w can be written as $w = L.u$ where L is language of a $\forall$-normal term and $u = r_1 \ldots r_m, r_i \in N_R$. iii) Set of paths of a word $w \in L(A)$, called $C(w)$, is defined as follows: $C((Q).\varepsilon)$=$\{Q\}$; $C((L(A_1) \cup L(A_2)).r) = \{C(L(A_1)).r \cup C(L(A_2)).r\}$. Since set of paths $C(w)$ does not contain structural informations, a word $w \in L(A)$ is not equivalent to the set of paths $C(w)$.
We now can characterize the semantics of a $\forall$-normal term using a tree which is defined as follows.

Definition 3.4. (Construction of normal term tree)
A is a $\forall$-*normal* term. The tree of the term A, T_A is defined as follows:

1. The root n_0 of T_A is created.

2. If A=Q or A=$Q_1 \sqcup Q_2$, respectively a node (Q) or (Q_1, Q_2) is created. The root n_0 is replaced by this node which becomes the root of T_A. If A=$\forall r.(Q_1 \sqcup Q_2)$, a node n_1=(Q_1, Q_2) is created. The tree T_A is constructed by adding an edge r in order to connect the node n_1 with n_0.

3. If A=$(A_1 \sqcup Q_1 \sqcup Q_2)$ and assume that the tree T_{A_1} of A_1 is built. Firstly, the root of T_{A_1} is replaced by the root node n_0. And then a node (Q_1, Q_2) which unifies all primitive concepts at top-level is created. This node is connected with the root node n_0 by an edge ε. If A=$A_1 \sqcup A_2$ and assume that the trees T_{A_1}, T_{A_2} are built. Two roots of these trees are unified in a unique node which is replaced by the root node n_0.

4. If $A = \forall r_1.(A_1 \sqcup A_2))$ and assume that two subtrees T_{A_1} and T_{A_2} are built. Firstly, two roots of the trees T_{A_1} and T_{A_2} are unified in a unique node n_1. And then the tree T_A is constructed by adding an edge r_1 in order to connect n_0 with n_1.

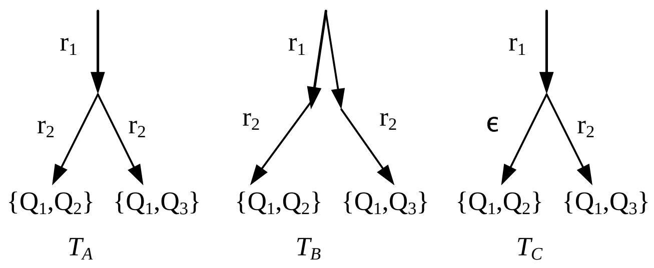

Figure 1: Trees T_X of $\forall$-normal terms

Remarks : i) In the construction of tree T_A, each leaf of tree T_A will be a set $\{Q_i\}$ i.e primitive concepts in $\{(Q_1), \ldots, (Q_m)\}.r$ are unified to a leaf $(Q_1, \ldots, Q_m)$.
ii) For the language L(A), the tree T_A which is constructed as in *Definition 3.4* is unique if the order of subtrees is not important.
iii) We denote $w \in C(A)$ as being a path from the root to a leaf of tree T_A and $P_A(w) = \{Q_i | Q_i$ belongs to the leaf $\{Q_i\}$of $w\}$. We write $W(A, Q)$ for set of paths of A terminated at Q in T_A.

Examples 3.1. (Fig. 1)
$A = \forall u_1.(\forall u_2.(P_1 \sqcup P_2) \sqcup (\forall u_2.(P_1 \sqcup P_3)))$, $L(A)$=$\{((P_1, P_2).u_2, (P_1, P_3).u_2).u_1\}$,
$C(A)$=$\{(P_1, P_2).u_2.u_1, (P_1, P_3).u_2.u_1\}$.
$B = \forall u_1.u_2.(P_1 \sqcup P_2) \sqcup \forall u_1.u_2.(P_1 \sqcup P_3)$ $L(B)$=$\{((P_1, P_2).u_2.u_1, (P_1, P_3).u_2.u_1\}$,
$C(B)$=$\{((P_1, P_2).u_2.u_1, (P_1, P_3).u_2.u_1\}$.
$C = \forall u_1.(P_1 \sqcup P_2 \sqcup (\forall u_2.(P_1 \sqcup P_3))$ $L(C)$=$\{((P_1, P_2).u_1, (P_1, P_3).u_2.u_1\}$,
$C(C)$=$\{((P_1, P_2).u_1, (P_1, P_3).u_2.u_1\}$.

Proposition 3.1. *Let T be an acyclic $\mathcal{ALU}$-TBox . Let I be a model of T. Let A be concept name occurring in T and be a $\forall$-normal term. For any $d \in \Delta^I$ (Δ^I=dom(I)), we can build a tree T_A^d w.r.t T_A i.e d is root of* *the tree and there exist individuals e_i corresponding to the nodes and leaves n_k of T_A and each edge $(e_i, e_j) = u_{ij}$ of T_A^d corresponds uniquely to an edge $(n_i, n_j) = u_{ij}$ of T_A. We have:*

$d \in A^I$ *iff*

for all trees T_A^d, there exists a path w of the tree T_A such that for all $e \in \Delta^I$: (d,e) $\in w^I \Rightarrow e \in Q_1^I \vee \ldots \vee e \in Q_m^I$ where $Q_i \in P_A(w)$.

A proof of this proposition can be found in [10]. Note that the semantics of an $\exists$-normal term in $\mathcal{ELU}$ can be similarly defined as follow :

Proposition 3.1.1. *Let T be an acyclic $\mathcal{ELU}$-TBox . Let I be a model of T. Let A be a concept name occurring in T and be a $\exists$-normal term. For any $d \in \Delta^I$ (Δ^I = dom(I)), we can build a tree T_A^d w.r.t T_A i.e d is root of the tree and there exist individuals e_i corresponding to the nodes and leaves n_k of T_A and each edge $(e_i,\ e_j) = u_{ij}$ of T_A^d corresponds uniquely to an edge $(n_i,\ n_j) = u_{ij}$ of T_A. We have:*

$d \in A^I$ *iff*

there exists a tree T_A^d such that for all paths w of the tree T_A there exists $e \in \Delta^I$: $(d, e) \in w^I \wedge e \in Q_1^I \wedge \ldots \wedge e \in Q_m^I$ where $Q_i \in P_A(w)$.

Therefore, the results obtained for $\mathcal{ALU}$ can be extended to $\mathcal{ELU}$.
There are at hand the notion of language of a $\forall$-normal term, we now can define an order relation on set of languages as the following.

Definition 3.5. *Let $L(A)$ and $L(B)$ be languages of $\forall$-normal terms.*

(I) *For all $L(A)$: $L(\bot) \preceq L(A)$, $L(A) \preceq L(\top)$.*
(II) *$L(Q_m \sqcup \ldots \sqcup Q_n) \preceq L(Q_p \sqcup \ldots \sqcup Q_q)$ iff $\{Q_m, \ldots, Q_n\} \subseteq \{Q_p, \ldots, Q_q\}$.*
(III) *$L(A) \preceq L(B)$ iff for all words m=$L_m.u \in L(A)$ there exists a word $n = L_n.v \in L(B)$ such that $u = v$ and $L_m \preceq L_n$.*
(IV) *$T_A \preceq T_B$ iff $L(A) \preceq L(B)$.*

From the partial relation that is defined on set of languages in *Definition 3.5*, we can characterize subsumption relation on set of $\forall$-*normal* terms.

Proposition 3.2. *Let T be an acyclic $\mathcal{ALU}$-TBox . Let I be a model of T. Let A_1 and A_2 be $\forall$-normal terms occurring in T, we have:*

$A_1 \sqsubseteq A_2$ *iff* $L(A_1) \preceq L(A_2)$.

A proof of *Proposition 3.2* can be found in [10]. In example 3.1, from Proposition 3.2 we have $B \sqsubseteq A$.

We now extend these results for the language $\mathcal{ALU}$. Firstly, we define language $L(C)$ and tree T_C of an $\mathcal{ALU}$-concept description C. This definition also allows us to determine $\forall$-normal form of an $\mathcal{ALU}$-concept description C.

Definition 3.6. *Let C be an $\mathcal{ALU}$-concept description.*

(I) *The $\forall$-normal form of C : $C = C_1 \sqcap \ldots \sqcap C_m$ where C_i are $\forall$-normal terms and all $\top, \bot$ are eliminated at top-level by the following rules : $P \sqcap \neg P \rightarrow \bot$, $E \sqcap \top \rightarrow E$ and $E \sqcap \bot \rightarrow \bot$.*
(II) $L(C_1 \sqcap \ldots \sqcap C_m) = \{L(C_1), \ldots, L(C_m)\}$.
(III) $T_{(C_1 \sqcap \ldots \sqcap C_m)} = \{T_{C_1}, \ldots, T_{C_m}\}$.

Remarks : i) According to the *Remarks* of Definition 3.2, an $\mathcal{ALU}$-concept description C can be always transformed into $\forall$-normal form. This transformation can take an exponential time in the size of C. ii)The semantics of C are naturally translated into the semantics of the set of trees T_{C_i}.

Theorem 3.1. *Let T be a acyclic $\mathcal{ALU} - TBox$. Let I be a model of T. Let A_1 and A_2 be concept descriptions occurring in T where $A_1 = A_{11} \sqcap \ldots \sqcap A_{1u}$, $A_2 = A_{21} \sqcap \ldots \sqcap A_{2v}$ are $\forall$-normal forms. We denote: $L(A_1) = \{L(A_{11}), \ldots, L(A_{1u})\}$, $L(A_2) = \{L(A_{21}), \ldots, L(A_{2v})\}$. We have :*

$$A_1 \sqsubseteq A_2 \text{ iff for all } L(A_{2j}) \in L(A_2), \text{ there exists } L(A_{1i}) \in L(A_1) \text{ such that } L(A_{1i}) \preceq L(A_{2j}).$$

A proof of *Theorem 3.1* can be found in [10]. Our problem in this section is, however, to find a $\mathcal{FL}_\neg$-approximation D of an $\mathcal{ALU}$-concept description C. *Corollary 3.1* will show that the subsumption characterization in language $\mathcal{ALU}$ is an extension of method using automata theory mentionned in [1]. This relationship allows us to compute the approximation $\mathcal{ALU}-\mathcal{FL}_\neg$.

Corollary 3.1. *Let C be L_s-concept description where $L_s = \mathcal{ALU}$ and $L_d = \mathcal{FL}_\neg$. There exists an upper approximation D of C in L_d and the computing of the approximation can take at most an exponential time in the size of C.*

Proof: We denote $\mathrm{T}_C = \{\mathrm{T}_{C_1}, \ldots, \mathrm{T}_{C_m}\}$ as a set of trees where C is considered as a conjunction of C_i terms and T_{C_i} corresponds to C_i. According to [1], each concept description D in $\mathcal{FL}_\neg$ is characterized by the words of $\mathrm{L}(D, Q_i)$ which can be considered as trees terminated by Q_i. Hence, we can build T_D from words of $\mathrm{L}(D, Q_i)$ for all Q_i. Consequently, we obtain a necessary and sufficient condition for the expressiveness of C in L_d from *Theorem 3.1* where it states that each T_{C_i} must be reduced to a path.
Existence and computing of the upper approximation D of C in L_d : $C \sqsubseteq D$.

Assume that there exist T_{C_k} such that T_{C_k} are paths. We denote the corresponding concept descriptions : $C_{k_1}, \ldots, C_{k_n}$. In this case, from *Theorem 3.1* the upper approximation D is built as a conjunction of C_{k_i}: $D = min^{\mathcal{ALU}}_{\mathcal{FL}_\neg} = \bigsqcap_{i=1}^{n} C_{k_i}$. If there does not exist T_{C_k} where T_{C_k} is a path, then $D = min^{\mathcal{ALU}}_{\mathcal{FL}_\neg} = \top$. The $\forall$-normalization of C needs at most an exponential time in the size of C. The research of path trees T_{C_k} takes also at most an exponential time in the size of C. Hence, whole computation can be carried out in exponential time in the size of C. ■

An algorithm can be built directly from *Corollary 3.1* which allows us to compute the upper approximation D in $\mathcal{FL}_\neg$ from a concept description C in $\mathcal{ALU}$ up to equivalence.

Example 3.2.
i) $min_{\mathcal{FL}_\neg}$(A) $= \top$; $min_{\mathcal{FL}_\neg}(A \sqcap \forall r.P_1 \sqcap \forall r.P_2)= \forall r.(P_1 \sqcap P_2)$ where A is defined in Example 3.1.

ii) Let $C= \forall r.P_1 \sqcap \forall r.(P_1 \sqcup \neg P_1) \sqcap \forall r.(P_2 \sqcap (P_3 \sqcup P_4))$.
Normalizing: $C= \forall r.P_1 \sqcap \forall r.\top \sqcap \forall r.P_2 \sqcap \forall r.(P_3 \sqcup P_4) = \forall r.P_1 \sqcap \forall r.P_2 \sqcap \forall r.(P_3 \sqcup P_4)$.
Finding path trees : $\forall r.P_1, \forall r.P_2$.
$min^{\mathcal{ALU}}_{\mathcal{FL}_\neg}(C)= \forall r.P_1 \sqcap \forall r.P_2$.

4 $\mathcal{ALU}$-$\mathcal{ALE}$ ($\mathcal{ELU}$-$\mathcal{ALE}$) and $\mathcal{ALC}$-$\mathcal{ALE}$ approximations

In this section, we propose an approach for computating the $\mathcal{ALE}$-approximation of a concept description containing disjunction. This approach is based on three events : i) computing the normal forms w.r.t conjunction and disjunction. ii) computing *lcs* (*least common subsumer*) of disjuncts on top-level. iii) propagating value restrictions on existential restrictions of conjuncts on top-level.
For the sake of simplicity, we assume that the set N_R of role names is $\{r\}$. However, all obtained results in this section can be generalized to arbitrary sets of role names.

Definition 4.1. (the following notations are used in [3]) *C is an $\mathcal{ALC}$-concept description* (c for conjunction, d for disjunction).

- *PRIM^c(C) or PRIM(C) (PRIM^d(C)) denotes the set of all (negated) concept names and the bottom (top) concept occurring on the top-level conjunction (disjunction) of C.*

- *$VAL_r^c(C)$ or $VAL_r(C)$ ($VAL_r^d(C)$) is conjunction (a set) of all C' occurring in value restrictions of the form $\forall r.C'$ on top-level of C. If there is no value restriction on top-level of C then VAL_r^c $(C) = \top$.*

- *$EX_r^d(C)$ ($EX_r^c(C)$ or $EX_r(C)$) is set (disjunction) of all C' occurring in existential restrictions of the form $\exists r.C'$ on top-level of C.*

- *The d-normal form of C (d-normal(C))*
  $C = C_1 \sqcup \ldots \sqcup C_m$ *where*
  $C_i = \sqcap_{A \in PRIM(Ci)} A \sqcap \bigsqcap_{C' \in EXr(Ci)} \exists r.C' \sqcap \forall r.\, VAL_r^c(C_i)$,
  $\perp \sqsubset C_i$, *C' and* $VAL_r^c(C_i)$ *are in d-normal forms.*

- *The c-normal form of C (c-normal(C))*
  $C = C_1 \sqcap \ldots \sqcap C_n$ *where*
  $C_i = \sqcup_{A \in PRIM(Di)} A \sqcup \exists r.EX_r^d(C_i) \sqcup \bigsqcup_{C' \in VALr(Di)} \forall r.C'$
  $C_i \sqsubset \top$, *C'and* $EX_r^d(C_i)$ *are in c-normal forms.*

Remarks: i) If C is an $\mathcal{ALC}$-concept description, C can be turned into an equivalent concept description of the *d-normal* form or an equivalent concept description of the *c-normal* form.
ii) *c-normal* form and *d-normal* form of an $\mathcal{ALC}$-concept description can be exponentially larger than C itself.
We now define *propagated normal form* of an $\mathcal{ALC}$-concept description. This is an extension of the *normal form* mentionned in [5] for $\mathcal{ALE}$-concept description. The propagated normal form of *c-normal* form of an $\mathcal{ALC}$-concept description allows us to compute the approximation $\mathcal{ALC}$-$\mathcal{ALE}$ as a conjunction of approximations on smaller terms.

Definition 4.2. *Let C be an $\mathcal{ALC}$-concept description. C is in propagated normal if none of the following rules can be applied at top-level in C_i :*

$$\begin{array}{rcl}
P \sqcap \neg P & \rightarrow & \perp \\
E \sqcap \perp & \rightarrow & \perp \\
\exists r.\perp & \rightarrow & \perp \\
\forall r.\top & \rightarrow & \top \\
E \sqcap \top & \rightarrow & \top \\
\forall r.E \sqcap \forall r.F & \rightarrow & \forall r.(E \sqcap F) \\
\forall r.E \sqcap \exists r.F & \rightarrow & \forall r.E \sqcap \exists r.(E \sqcap F)
\end{array}$$

where C_i are terms occurring in d-normal form of $C = C_1 \sqcup \ldots \sqcup C_m$.
Remarks. i) Propogated normal transformation increases polynomially the size of the d-normal form of C.

Definition 4.3. *Let C_1, ... ,C_n be the L_{-}concept descriptions. The L_{-}concept description C is the least common subsumer (lcs) of C_1, ... ,C_n (C = lcs(C_1,*

...,C_n) for short) iff i) $C_i \sqsubseteq C$ for all i=1..n ii) $C_i \sqsubseteq C'$ for all i=1..n that implies $C \sqsubseteq C'$.

The main idea of the efficient algorithm is based on the possibility of distributing approximation computation over disjuncts and conjuncts. However, the distribution only holds for disjuncts owing to the application of *lcs* on the distribution. A further normalization is required for conjunctions before the distribution. In fact, the propagated normal form allows the distribution to hold for conjunction. This property is guaranteed by the following proposition.

Proposition 4.1.

1. *If $A = A_1 \sqcap C$ and $B = B_1 \sqcap C$ are propagated $\mathcal{ALE}$-concept descriptions i.e the rules in Definition 4.2 are applied to $\mathcal{ALE}$-concept descriptions, we have : $lcs(A,\ B) = C \sqcap\ lcs(A_1,\ B_1)$.*

2. *If $C = C_1 \sqcup C_2$ where C is an $\mathcal{ALC}$-concept description and $\bot \sqsubset C_1, C_2$ then, $min_{\mathcal{ALE}}(C)\ = lcs(min_{\mathcal{ALE}}(C_1), min_{\mathcal{ALE}}(C_2))$.*

3. *If $C = C_1 \sqcap C_2$ is a propagated normal $\mathcal{ALC}$-concept description, we have : $min_{\mathcal{ALE}}(C)\ = min_{\mathcal{ALE}}(C_1) \sqcap\ min_{\mathcal{ALE}}(C_2)$.*

A proof of *Proposition 4.1* can be found in [10]. Naturally, we immediately have an algorithm computing approximation $\mathcal{ALU}$-$\mathcal{ALE}$:

Input: $\mathcal{ALU}$-concept description C Output: $min_{\mathcal{ALE}}(C)$

- If $C = \forall r.C_1$ ($C = \exists r..C_1$) , $min_{\mathcal{ALE}}(C) = \forall r.(min_{\mathcal{ALE}}(C_1)\ )$ ($min_{\mathcal{ALE}}(C) = \exists r.(min_{\mathcal{ALE}}(C_1))$)
- If C can be written in $C = C_1 \sqcup C_2$ where $\bot \sqsubset C_1,\ C_2$, $min_{\mathcal{ALE}}(C) = lcs(min_{\mathcal{ALE}}(C_1),\ min_{\mathcal{ALE}}(C_2))$
- If $C = C_1 \sqcap C_2$ where $C_1,\ C_2 \sqsubset \top$, $min_{\mathcal{ALE}}(C) = min_{\mathcal{ALE}}(C_1) \sqcap\ min_{\mathcal{ALE}}(C_2)$

Figure 2: Algorithm for $\mathcal{ALU}$($\mathcal{ELU}$)-$\mathcal{ALE}$ approximation

Theorem 4.1. *Let C be an $\mathcal{ALU}$ ($\mathcal{ELU}$)-concept description.*

1. *If $C \equiv \bot$ or $C \equiv \top$ then $min_{\mathcal{ALE}}(C) = \bot$ or $min_{\mathcal{ALE}}(C) = \top$.*

2. *If* $C = C_1 \sqcup C_2$ *where* $\bot \sqsubset C_1,\ C_2$ *then,* $min_{\mathcal{ALE}}(C)\ =\ lcs(min_{\mathcal{ALE}}(C_1),\ min_{\mathcal{ALE}}(C_2))$.

3. *If* $C = C_1 \sqcap C_2$ *where* $C_1,\ C_2 \sqsubset \top$, *then* $min_{\mathcal{ALE}}(C)\ =\ min_{\mathcal{ALE}}(C_1) \sqcap\ min_{\mathcal{ALE}}(C_2)$.

A proof of *Theorem 4.1* can be found in [10].

Corollary 4.1. *The algorithm* $min^{\mathcal{ALC}}_{\mathcal{ALE}}(C)$ *takes at most an exponential time in the size of* C *where* C *is an* $\mathcal{ALU}(\mathcal{ALE})$*-concept description.*
A priori, the algorithm that is introduced in *Theorem 4.1* can be extended to approximation $\mathcal{ALC}$-$\mathcal{ALE}$. However, $\mathcal{ALC}$ is not a propagated normal language by default. Therefore, so that $\mathcal{ALC}$ becomes propagated normal, further transformations are required. *Theorem 4.2.* shows how this process operates.

Theorem 4.2. *Let* C *be an* $\mathcal{ALC}_{-}$*concept description.*

1. *If* $C \equiv \bot$ *or* $C \equiv \top$ *then* $min_{\mathcal{ALE}}(C) = \bot$ *or* $min_{\mathcal{ALE}}(C) = \top$.

2. *If* $C = C_1 \sqcup C_2$ *where* $\bot \sqsubset C_1,\ C_2$ *then,* $min_{\mathcal{ALE}}(C)\ =\ lcs(min_{\mathcal{ALE}}(C_1),\ min_{\mathcal{ALE}}(C_2))$.

3. *If* $C = C_1 \sqcap \ldots \sqcap C_m$ *where* $C_i \sqsubset \top$. *We have* $min_{\mathcal{ALE}}(C) = \bigsqcap_q min_{\mathcal{ALE}}(E_q)$ *where the size of* E_i *is a polynomial in the size of* C *and the number of terms* E_i *is exponential in the size of* C.

A proof of *Theorem 4.2* can be found in [10].

Corollary 4.2. *The algorithm* $min^{\mathcal{ALC}}_{\mathcal{ALE}}(C)$ *takes an exponential time in the size of* C *where* C *is an* $\mathcal{ALC}$*-concept description.*
A proof of *Corollary 4.2* can be found in [10].

Example 4.1.
$C = \forall r.B \sqcap \exists r.A \sqcap\ (\exists r.B \sqcup\ \forall r.B)$.

(Step 3.1) Computing *d-normal* of C : $(\forall r.B \sqcap \exists r.A \sqcap \exists r.B) \sqcup (\forall r.B \sqcap \exists r.A)$
(Step 3.2) Propagating *d-normal* : $(\forall r.B \sqcap \exists r.(A \sqcap B) \sqcap\ \exists r.B) \sqcup (\forall r.B \sqcap \exists r.(A \sqcap B))$
(Step 2.) Computing *c-normal* : $\forall r.B \sqcap\ (\forall r.B \sqcup \exists r.(A \sqcap B)) \sqcap\ (\exists r.(A \sqcap B) \sqcup \forall r.B) \sqcap \exists r.(A \sqcap B) \sqcap\ (\exists r.B \sqcup \forall r.B) \sqcap\ (\exists r.B \sqcup \exists r.(A \sqcap B))$
(Step 3.4) Computing the approximation:
min(C) = min($\forall r.B$)$\sqcap$ min($\forall r.B \sqcup \exists r.(A \sqcap B)$) $\sqcap$
min($\exists r.(A \sqcap B) \sqcup \forall r.B$)$\sqcap$ min($\exists r.(A \sqcap B)$) $\sqcap$ min($\exists r.B \sqcup \forall r.B$)
$\sqcap$ min($\exists r.B \sqcup \exists r.(A \sqcap B)$)

$= \forall r.B \sqcap$ lcs($\forall r.B$, $\exists r.(A \sqcap B)$)$\sqcap$ lcs($\exists r.(A \sqcap B)$, $\forall r.B$)$\sqcap$
min($\exists r.(A \sqcap B)$)$\sqcap$ lcs($\exists r.B$, $\forall r.B$)$\sqcap$ min($\exists r.B \sqcup \exists r.(A \sqcap B)$)
$= \forall r.B \sqcap \top \sqcap \top \sqcap \exists r.(A \sqcap B) \sqcap \top \sqcap \exists r.B = \forall r.B \sqcap \exists r.(A \sqcap B) \sqcap$
$\exists r.B$ ■

Input: $\mathcal{ALC}$-concept description C
Output: $min_{\mathcal{ALE}}(C)$

1. If $C = \forall r.C_1$ ($C = \exists r.C_1$) , $min_{\mathcal{ALE}}(C) = \forall r.(min_{\mathcal{ALE}}(C_1)$ ($min_{\mathcal{ALE}}(C) = \exists r.(min_{\mathcal{ALE}}(C_1))$
2. If C can be written in $C = C_1 \sqcup ... \sqcup C_n$ where $\bot \sqsubset C_i$
   $min_{\mathcal{ALE}}(C) = lcs(min_{\mathcal{ALE}}(C_1)$,..., $min_{\mathcal{ALE}}(C_n))$
3. If C can be written in $C = C_1 \sqcap ... \sqcap C_n$ where $C_i \sqsubset \top$
   3.1 Computing *d-normal* form of C : *d-normal*(C)= $D_1 \sqcup \ldots \sqcup D_n$
   3.2 For each D_i, replacing $\exists r.C$' with $\exists r.(C' \sqcap VAL_r(D_i))$, where C' $\in EX_r(D_i)$, we obtain *d-normal*(C)= $\bigsqcup_i D_i^{'}$
   3.3 Computing *c-normal*(*d-normal*(C)) $= \bigsqcap_1^q E_i$
   3.4 $min_{\mathcal{ALE}}(C) = \bigsqcap_1^q min_{\mathcal{ALE}}(E_i)$

Figure 3: Algorithm for $\mathcal{ALC}$-$\mathcal{ALE}$ approximation

5 Conclusion

We have proposed a way of characterizing subsumption in $\mathcal{ALU}$. This way can be extended for $\mathcal{ELU}$. From the subsumption characterization, we have proposed an exponential algorithm in the worst case for approximating an $\mathcal{ALU}$-concept description with $\mathcal{FL}_\neg$-concept description. We have also introduced an efficient algorithm for computing upper approximation of an $\mathcal{ALC}$-concept description with $\mathcal{ALE}$-concept description. This algorithm runs in exponential time in the size of the $\mathcal{ALC}$-concept description. At present, we know that only structural approach allows us to develop non-standard inference services. An interesting question is how to use structural approach for subsumption in $\mathcal{ALC}$ in order to extend approximation inference to expressive languages. On the other hand, in electronic commerce the interpretation of received terms depends on context informations. It remains to be seen how we may integrate context informations into ontologies. Our work in the future aims to address these questions.

References

[1] Franz Baader. Using automata theory for characterizing the semantics of terminological cycles. *Annals of Mathematics and Artificial Intelligence*, 1996.

[2] Franz Baader, Ralf Küsters, and Ralf Molitor. Rewriting Concepts Using Terminologies. *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning* (KR2000).

[3] Sebastian Brandt, Ralf Küsters, Anni-Yasmin Turhan. Approximation and Difference in Description Logics. *Proceedings of the Eight International Conference on Principles of Knowledge Representation and Reasoning* (KR2002).

[4] Ralf Küsters. Non-standard Inferences in Description Logics. *Thesis, Springer* (2001)

[5] Franz Baader, Ralf Küsters, and Ralf Molitor. Computing least common subsumer in description logics with existential restrictions. *Proceedings of the 17th International Joint Conference Artificial Intelligence* (IJCAI 1999)

[6] Catriel Beeri, Alon Y. Levy and Marie-Christine Rousset. Rewriting Queries Using Views in Description Logics. *Proceedings of the 16th ACM Symposium on Principles of Database Systems (PODS-97), Tucson* (May 1997)

[7] F.M Donini, M. Lenzerini, D. Nardi. Reasoning in Description Logics. *CSLI Publications* (1997)

[8] M. Schmidt-Schauss and D.Smolka. Attributive Concept Descriptions with Complements. *Artificial Intelligence* (1991)

[9] Bernhard Nebel. Reasoning and revision in hybrid representation systems. *Thesis, Springer-Verlag* (1990)

[10] Chan Le Duc, Nhan Le Thanh. Approximation from a Description Logic with disjunction to another without disjunction. *Technical Report 2002.* See http://www.i3s.unice.fr/~cleduc.

[11] Chan Le Duc, Nhan Le Thanh. Problématique de l'ebXML et Logique de Description. *Technical Report 2002.* See http://www.i3s.unice.fr/~cleduc.