TECHNISCHE UNIVERSITÄT DRESDEN

INTERNATIONAL CENTER FOR COMPUTATIONAL LOGIC

Transforming Fuzzy Description Logic $\mathcal{ALC}_{\mathcal{FL}}$ into Classical Description Logic $\mathcal{ALCH}$

Yining Wu

Overseeing Professor: Prof. Steffen Hölldobler
Supervisor: Dr. habil. Pascal Hitzler

Department of Computer Science
Artificial Intelligence Institute
TECHNISCHE UNIVERSITÄT DRESDEN
Germany March, 2007

TECHNISCHE UNIVERSITÄT DRESDEN

Author:	**Yining Wu**
Matrikel-Nr.:	**3085253**
Title:	**Transforming Fuzzy Description Logic $\mathcal{ALC}_{\mathcal{FL}}$ into Classical Description Logic $\mathcal{ALCH}$**
Degree:	**Master of Science**
Date of submission:	**30 March 2007**

Declaration

Hereby I certify that the thesis has been written by me. Any help that I have received in my research work has been acknowledged. Additionally, I certify that I have not used any auxiliary sources and literature except those I cited in the thesis.

Signature of Author

Acknowledgments

I would like to express my gratitude to my supervisor, Dr. habil. Pascal Hitzler of Institute AIFB, Universität Karlsruhe who gave me the chance to do this thesis for his advice, encouragement, enduring patience and constant support.

I also wish to thank Prof. Steffen Hölldobler of the Computer Science Department at Technische Universität Dresden. He graciously volunteered to act as my supervisor since Dr. habil. Pascal Hitzler works in Karlsruhe. His suggestions, comments and guidance were invaluable to the completion of this work.

Additionally, I want to thank Nguyen Hoàng Ngà who was helpful in providing the additional guidance and expertise I needed in order to accomplish this work.

Finally, I must thank my dear parents. They always stand behind me with continuing, loving and financial support without complaint. I can't finish this work without their support.

Abstract

We consider $\mathcal{ALC}_{\mathcal{FL}}$ [5] in this thesis. $\mathcal{ALC}_{\mathcal{FL}}$ is a fuzzy Description Logic with hedges as its fuzzy extension. Three example scenarios of $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base are shown. The linear symmetric hedge algebra [4, 9, 10] is introduced in order to represent the truth domain of interpretations of the $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base. The aim of this thesis is to present a satisfiability preserving transformation of $\mathcal{ALC}_{\mathcal{FL}}$ into the description logic $\mathcal{ALCH}$, i.e., the extension of $\mathcal{ALC}$ by role hierarchies. The soundness and completeness of this transformation are proved. A contribution of this transformation is that we can do reasoning of $\mathcal{ALC}_{\mathcal{FL}}$ by using already existing Description Logic systems. The transformation is implemented in both the programming language JAVA and Prolog. Testing is done in JAVA using APIs of Jena,[1] KAON2[2] and Racer.[3]

[1]http://jena.sourceforge.net/
[2]http://kaon2.semanticweb.org/
[3]http://www.racer-systems.com/

Contents

Chapter 1

Introduction

1.1 An introduction to Description Logics

Description Logics (DLs [1]) are a family of knowledge representation languages that can be used to represent the knowledge of an application domain in a structured and formally well-understood way. DLs are equipped with a formal, logic-based semantics. Knowledge bases (KBs) can be set up by a knowledge representation (KR) system which is based on DLs. A KB consists of two components, the TBox and the ABox. The TBox introduces the *terminology*, i.e., the vocabulary of an application domain, while the ABox contains *assertions* about named individuals in terms of this vocabulary. KR systems also provide facilities to do reasoning about the content of KBs.

Reasoning is a central service of DLs. We can deduce implicitly represented knowledge from the knowledge that is explicitly contained in the knowledge base by reasoning. DLs support classification of concepts and individuals. Classification of concepts is called subsumption relationships in DLs which determines subconcept/superconcept relationships between concepts by that we can build a hierarchy structure to provide information on the connection between concepts. Classification of individuals specifies whether an individual is an instance of a concept in terms of the description of the individual and the definition of the concept. Useful information about the properties of an individual can be obtained.

DLs are the basis of the web ontology language OWL.[1] OWL is intended to provide a language that can be used to describe the classes and relations between them that are inherent in web documents and applications. A concept in DL is referred to as a class in OWL. A role in DL is a property in OWL.

1.2 Motivation

The concepts in classical DLs are usually interpreted as crisp sets, i.e., an individual either belongs to the set or not. In the real world, the answers to some questions are

[1] Please visit http://www.w3.org/TR/owl-guide/ for more details

often not only yes or no, rather we may say that an individual is an instance of a concept only to some certain degree. For instance, $Young$ is a concept for which we can usually not fix a boundary such that a person is old if he is older than the boundary otherwise he is young. Actually the person is not old if he is only one year older than that boundary. We prefer saying that an individual Tom is an instance of the concept $Young$ to some certain degree $n \in [0, 1]$ depending on Tom's age to just saying Tom is young or not. Classical DLs whose semantics is based on classical first-order logic cannot express vague or uncertain knowledge. To overcome this deficiency, approaches for integrating fuzzy logic into DLs have been proposed. Fuzzy DLs can directly handles the notion of vagueness and imprecision.

In [13] Umberto Straccia presents a quite general fuzzy DL, $\mathcal{ALC}_{\mathcal{F}}$ which is based on the DL $\mathcal{ALC}$ (stands for Attributive Language with Complement) [1], a significant and expressive representative of the various DLs. $\mathcal{ALC}$ is an extension of $\mathcal{AL}$ which is introduced in [12] as a minimal language that is of practical interest. From a computational complexity point of view, the additional expressive power has no impact on the complexity of reasoning in $\mathcal{ALC}_{\mathcal{F}}$. This is certainly important as the nice trade-off between computational complexity and expressive power of DLs contributes to their popularity. In $\mathcal{ALC}_{\mathcal{F}}$, ABoxes are equipped with degrees. We can express that an individual is an instance of a concept with a certain degree which is represented by a real number in the interval $[0, 1]$, e.g., Tom is Young with the degree which is greater than 0.8.

In real life, people use natural language to think, to reason, to deduce conclusions, and to make decisions. It is more intuitional to say Tom is very young or, alternatively, that Tom is $Young$ with degree $VeryTrue$ than to say Tom is young to the degree 0.8. Fuzzy logics provides different options for fuzzy extensions. The fuzzy DL $\mathcal{ALC}_{\mathcal{FL}}$ is presented in [5] which uses adverbs (or hedges) such as "very","possibly" and "less" to express the fuzzy values instead of using real numbers.

The following table lists some fuzzy DLs. $\mathcal{ALC}_{\mathcal{F}}$, $\mathcal{ALC}_{\mathcal{FH}}$ and $\mathcal{ALC}_{\mathcal{FLH}}$ use real numbers to represent fuzzy values, while $\mathcal{ALC}_{\mathcal{FL}}$ uses hedges.

DL	truth domain	syntax
$\mathcal{ALC}_{\mathcal{F}}$	$[0, 1]$	$A, \neg C, C \sqcap D, C \sqcup D, \forall R.C, \exists R.C$
$\mathcal{ALC}_{\mathcal{FH}}$	$[0, 1]$	$A, MA, \neg C, C \sqcap D, C \sqcup D, \forall R.C, \exists R.C$
$\mathcal{ALC}_{\mathcal{FLH}}$	$[0, 1]$	$A, MC, MR, \neg C, C \sqcap D, C \sqcup D, \forall R.C, \exists R.C$
$\mathcal{ALC}_{\mathcal{FL}}$	HAs	$A, MC, \neg C, C \sqcap D, C \sqcup D, \forall R.C, \exists R.C$

$\mathcal{ALC}_{\mathcal{F}}$ [13] is an extension of the classical DL $\mathcal{ALC}$. $\mathcal{ALC}_{\mathcal{FH}}$ [6], where primitive concepts are modified by means of hedges, is strictly more expressive than $\mathcal{ALC}_{\mathcal{F}}$. $\mathcal{ALC}_{\mathcal{FLH}}$ [11] is based on $\mathcal{ALC}_{\mathcal{FH}}$, but linear hedges are used instead of exponential ones. In $\mathcal{ALC}_{\mathcal{FLH}}$, roles and arbitrary concepts can be modified by hedges. This extends $\mathcal{ALC}_{\mathcal{FH}}$ where only primitive concepts can be modified. $\mathcal{ALC}_{\mathcal{FL}}$ uses Hedge Algebras (HAs) [4] to represent the truth domain of interpretations.

As we know, one feature of classical DLs is the emphasis on reasoning as a central service. The fuzzy DLs have this feature as well, but there are few reasoners for fuzzy DLs. Transforming fuzzy DLs into classical ones makes the reasoning in fuzzy DLs feasible using already existing DL systems and take advantages of their optimizations to achieve efficiency. In this work we present a satisfiability preserving transformation of $\mathcal{ALC}_{\mathcal{FL}}$ into $\mathcal{ALCH}$.

1.3 The structure of the thesis

Chapter 2 introduces the formal foundation necessary for the subsequent presentation of our transformation. In Section 2.1 the syntax and semantics of $\mathcal{ALCH}$ are defined. In Section 2.2 we review the basic notions of linear symmetric hedge algebras, extend the order relation to use it as truth domain and then define the inverse mapping of hedges. Section 2.3 provides definitions of the syntax and semantics of $\mathcal{ALC}_{\mathcal{FL}}$. We give some examples in Section 2.4 to show why we need fuzzy DLs with hedges.

Chapter 3 presents the transformation of $\mathcal{ALC}_{\mathcal{FL}}$ into $\mathcal{ALCH}$ and the theorem which claims that the transformation is satisfiability preserving. The proof of the theorem in Section 3.4 needs two Lemmas which are also proved in the same Section.

In Chapter 4 we illustrate the implementation of the reduction using JAVA, Jena and API of KAON2. We compare KAON2 and Racer by measuring the computing time of reasoning the output of the transformation. We talk about conclusion and the complexity of the transforming algorithm in the Chapter 5. At last, we give an example and show the result of the transformation using the example knowledge base as the input.

Chapter 2

Preliminaries

2.1 $\mathcal{ALCH}$

Elementary descriptions are concept names and role names. Complex descriptions can be built from them inductively with concept constructors. We consider the language $\mathcal{ALCH}$ (Attributive Language with Complement and role Hierarchy). In abstract notation, we use the letters A and B for concept names, the letter R for role names, and the letters C and D for concept terms.

Definition 1 (Syntax of $\mathcal{ALCH}$ concept terms). *Let N_R and N_C be disjoint sets of role names and concept names. Let $A \in N_C$ and $R \in N_R$. Concept terms in $\mathcal{ALCH}$ are formed according to the following syntax rule:*

$$
\begin{array}{lcll}
C, D & \longrightarrow & A| & \textit{(concept name)} \\
 & & \top| & \textit{(top concept)} \\
 & & \bot| & \textit{(bottom concept)} \\
 & & C \sqcap D| & \textit{(concept conjunction)} \\
 & & C \sqcup D| & \textit{(concept disjunction)} \\
 & & \neg C| & \textit{(concept negation)} \\
 & & \forall R.C| & \textit{(universal quantification)} \\
 & & \exists R.C| & \textit{(existential quantification)}
\end{array}
$$

The semantics of concept terms are defined formally by interpretations.

Definition 2 (Semantics of $\mathcal{ALCH}$ concept terms). *An* interpretation $\mathcal{I}$ *is a pair* $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$*, where* $\Delta^{\mathcal{I}}$ *is a nonempty set (*interpretation domain*) and* $\cdot^{\mathcal{I}}$ *is an* interpretation function *which assigns to each concept name A a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and to each role name R a binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The interpretation of complex concept terms is extended by the following inductive definitions:*

$$\begin{array}{rcl}
\top^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \\
\bot^{\mathcal{I}} & = & \emptyset \\
(C \sqcap D)^{\mathcal{I}} & = & C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} & = & C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(\forall R.C)^{\mathcal{I}} & = & \{d \in \Delta^{\mathcal{I}} \mid \forall d'.(d,d') \notin R^{\mathcal{I}} \textit{ or } d' \in C^{\mathcal{I}}\} \\
(\exists R.C)^{\mathcal{I}} & = & \{d \in \Delta^{\mathcal{I}} \mid \exists d'.(d,d') \in R^{\mathcal{I}} \textit{ and } d' \in C^{\mathcal{I}}\}
\end{array}$$

A concept term C is *satisfiable* iff there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$, denoted by $\mathcal{I} \models C$. Two concept terms C and D are *equivalent* (denoted by $C \equiv D$) iff $C^{\mathcal{I}} = D^{\mathcal{I}}$ for all interpretation $\mathcal{I}$.

We have seen how we can form complex descriptions of concepts to describe classes of objects. Now, we introduce *terminological axioms*, which make statements about how concept terms and roles are related to each other respectively.

In the most general case, *terminological axiom* have the form

$$C \sqsubseteq D \text{ or } R \sqsubseteq S,$$

where C, D are concept terms, R, S are role names. This kind of terminological axioms are also called *inclusions*. A set of axioms of the form $R \sqsubseteq S$ is called *role hierarchy*. An interpretation $\mathcal{I}$ *satisfies* an inclusion $C \sqsubseteq D$ ($R \sqsubseteq S$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ ($R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$), denoted by $\mathcal{I} \models C \sqsubseteq D$ ($\mathcal{I} \models R \sqsubseteq S$).

Definition 3 (TBox). *A* terminology*, i.e.,* TBox*, is a finite set of terminological axioms.*

An interpretation $\mathcal{I}$ *satisfies* (is a *model* of) a terminology $\mathcal{T}$ iff $\mathcal{I}$ *satisfies* each element in $\mathcal{T}$, denoted by $\mathcal{I} \models \mathcal{T}$.

Assertions define how individuals relate with each other and how individuals relate with concept terms.

Definition 4 (Assertion). *Let N_I be a set of individual names which is disjoint to N_R and N_C. An* assertion *α is an expression of the form,*
- $a{:}C$ *(concept assertion) or*
- $(a,b){:}R$ *(role assertion),*

where $a, b \in N_I$, $R \in N_R$ and $C \in N_C$.

An interpretation $\mathcal{I}$ *satisfies* a concept assertion $a : C$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$, denoted by $\mathcal{I} \models a : C$. $\mathcal{I}$ *satisfies* a role assertion $(a, b) : R$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$, denoted by $\mathcal{I} \models (a, b) : R$.

Definition 5 (ABox). *A finite set of* assertions *is called* ABox*.*

An interpretation $\mathcal{I}$ *satisfies* (is a *model* of) an ABox $\mathcal{A}$ iff $\mathcal{I}$ *satisfies* each assertion in $\mathcal{A}$, denoted by $\mathcal{I} \models \mathcal{A}$.

Definition 6 (Knowledge Base). *A* knowledge base *is of the form $\langle \mathcal{T}, \mathcal{A} \rangle$ where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox.*

Example 7. *A KB* $\mathcal{K} = \langle\{A \sqsubseteq \forall R.\neg B\}, \{a : \forall R.C\}\rangle$.

An interpretation $\mathcal{I}$ *satisfies* (is a *model* of, denoted by $\mathcal{I} \models \mathcal{K}$) a knowledge base $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$ iff $\mathcal{I}$ *satisfies* both $\mathcal{T}$ and $\mathcal{A}$. We say that a knowledge base $\mathcal{K}$ *entails* an assertion α, denoted $\mathcal{K} \models \alpha$ iff each model of $\mathcal{K}$ satisfies α. Furthermore, let $\mathcal{T}$ be a TBox and let C, D be two concept terms. We say that D *subsumes* C with respect to $\mathcal{T}$ (denoted by $C \sqsubseteq_{\mathcal{T}} D$) iff for each model of $\mathcal{T}$, $\mathcal{I} \models C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.

The problem of determining whether $\mathcal{K} \models \alpha$ is called *entailment problem*; the problem of determining whether $C \sqsubseteq_{\mathcal{T}} D$ is called *subsumption problem*; and the problem of determining whether $\mathcal{K}$ is satisfiable is called *satisfiability problem*. Entailment problem and subsumption problem can be reduced to satisfiability problem.

2.2 Hedge Algebra

In daily life, we often say linguistic terms such as "Very True" and "More or Less false" which are linguistic values of a linguistic variable "TRUTH" where "True" and "false" are vague concepts of "TRUTH". Adverbs as "Very", "More or Less" and "Possibly" are called hedges in fuzzy DLs. We observe that the set of linguistic values, or the domain of a linguistic variable, can be represented as a formal algebra with operations being hedges and generators being the vague concepts of this linguistic variable. Furthermore, according to the meaning of hedges, linguistic values can be partially ordered, e.g., very true > true. Based on some properties of hedges, we introduce the hedge algebra and the ordering relationship between vague concepts of a linguistic variable. We also explain how the hedge algebra represents the truth domain of interpretations of $\mathcal{ALC}_{\mathcal{FL}}$ knowledge bases.

2.2.1 Linear symmetric Hedge Algebra

In this section, we introduce linear symmetric Hedge Algebras (HAs). For general HAs, please refer to [4, 9, 10].

Let us consider a linguistic variable $TRUTH$ with the domain $dom(TRUTH) = \{True, False, VeryTrue, VeryFalse, MoreTrue, MoreFalse, PossiblyTrue, Possibly False \ldots\}$. This domain is an infinite partially ordered set, with a natural ordering $a < b$ meaning that b describes a larger degree of truth if we consider $True > False$. This set is generated from the basic elements (*generators*) $G = \{True, False\}$ by using *hedges*, i.e., unary operations from a finite set $H = \{Very, Possibly, More\}$. The $dom(TRUTH)$ which is a set of linguistic values can be represented as $X = \{\delta c \mid c \in G, \delta \in H^*\}$, From the algebraic point of view, the truth domain can be described as an abstract algebra $AX = (X, G, H, >)$.

To define relations between hedges, we introduce some notations first. We define that $H(x) = \{\sigma x \mid \sigma \in H^*\}$ for all $x \in X$. Let I be the identity hedge, i.e., $\forall x \in X.Ix = x$. The identity I is the least element. Each element of H is an *ordering operation*, i.e., $\forall h \in H$, $\forall x \in X$, either $hx > x$ or $hx < x$.

Definition 8. *Let* $h, k \in H$ *be two hedges, for all* $x \in X$ *we define:*

- h, k *are* converse *if* $hx < x$ *iff* $kx > x$;
- h, k *are* compatible *if* $hx < x$ *iff* $kx < x$;
- h *modifies terms stronger or equal than* k, *denoted by* $h \geq k$ *if* $hx \leq kx \leq x$ *or* $hx \geq kx \geq x$;
- $h > k$ *if* $h \geq k$ *and* $h \neq k$;
- h *is* positive wrt k *if* $hkx < kx < x$ *or* $hkx > kx > x$;
- h *is* negative wrt k *if* $kx < hkx < x$ *or* $kx > hkx > x$.

$ALC_{\mathcal{FL}}$ only considers symmetric HAs, i.e., there are exactly two generators as in the example $G = \{True, False\}$. Let $G = \{c^+, c^-\}$ where $c^+ > c^-$. c^+ and c^- are called *positive* and *negative generators* respectively. Because there are only two generators, the relations presented in Definition 8 divides the set H into two subsets $H^+ = \{h \in H \mid hc^+ > c^+\}$ and $H^- = \{h \in H \mid hc^+ < c^+\}$, i.e., every operation in H^+ is converse w.r.t. any operation in H^- and vice-versa, and the operations in the same subset are compatible with each other.

Definition 9 (Linear Symmetric Hedge Algebra). *An abstract algebra* $AX = (X, G, H, >)$, *where* $H \neq \emptyset, G = \{c^+, c^-\}$ *and* $X = \{\sigma c \mid c \in G, \sigma \in H^*\}$ *is called a* linear symmetric hedge algebra *if it satisfies the properties* (A1)-(A5).

(A1) Every hedge in H^+ is a converse operation of all operations in H^-.

(A2) Each hedge operation is either positive or negative w.r.t. the others, including itself.

(A3) The sets $H^+ \cup \{I\}$ and $H^- \cup \{I\}$ are linearly ordered with the least element I.

(A4) If $h \neq k$ and $hx < kx$ then $h'hx < k'kx$, for all $h, k, h', k' \in H$ and $x \in X$.

(A5) If $u \notin H(v)$ and $u \leq v$ ($u \geq v$) then $u \leq hv$ ($u \geq hv$), for any hedge h.

Let $AX = (X, G, H, >)$ be a linear symmetric hedge algebra and $c \in G$. We define that,

$$\bar{c} = \begin{cases} c^+ & \text{if} \quad c = c^- \\ c^- & \text{if} \quad c = c^+ \end{cases}$$

Definition 10 (Contradictory Element in Linear Symmetric HA). *Let* $AX = (X, G, H, >)$ *be a linear symmetric hedge algebra, where* $G = \{c^+, c^-\}$. *Let* $x \in X$ *and* $x = \sigma c$, *where* $\sigma \in H^*$ *and* $c \in G$. *The* contradictory element *to* x *is* $y = \sigma\bar{c}$ *written* $y = -x$.

[4] gave us the following proposition to compare elements in X.

Proposition 11. *Let $AX = (X, G, H, >)$ be a linear symmetric HA, $x = h_n \cdots h_1 u$ and $y = k_m \cdots k_1 u$ are two elements of X where $u \in X$. Then there exists an index $j \leq \min\{n, m\} + 1$ such that $h_i = k_i$ for all $i < j$, and*

(i) *$x < y$ iff $h_j x_j < k_j x_j$, where $x_j = h_{j-1} \cdots h_1 u$;*

(ii) *$x = y$ iff $n = m = j$ and $h_j x_j = k_j x_j$.*

Because $H^+ \cup \{I\}$, $H^- \cup \{I\}$ and G are linearly ordered, [4] presented the following proposition.

Proposition 12. *For a linear symmetric HA $AX = (X, G, H, >)$, X is* linearly ordered.

For $x, y \in X$ we define that $x \wedge y = \min(x, y)$ and $x \vee y = \max(x, y)$. Now we can introduce an implication operation : $x \to y = -x \vee y$.

If H is not empty then X are infinite, according to [10], the greatest element and the least element of X are $\sup(c^+)$ and $\inf(c^-)$ respectively, and $\inf(c^+) = \sup(c^-)$. We use W to represent $\inf(c^+)$ and $\sup(c^-)$, use 1 and 0 to represent $\sup(c^+)$ and $\inf(c^-)$ respectively. The following proposition([10]) shows that each linear symmetric hedge algebra can be taken as a logical foundation for reasoning methods:

Proposition 13. *For every symmetric extended HA, the following properties hold:*

1. $-hx = h(-x)$, *for any* $h \in H$;
2. $- - x = x$; $-1 = 0$, $-0 = 1$, *and* $-W = W$;
3. $-(x \vee y) = (-x \wedge -y)$ *and* $-(x \wedge y) = (-x \vee -y)$;
4. $x \wedge -x < W < y \vee -y$;
5. $x > y$ *iff* $-x < -y$;
6. $x \to y = -y \to -x$;
7. $x \to (y \to z) = y \to (x \to z)$;
8. $x \to y > x' \to y'$ *if* $x < x'$ *and/or* $y > y'$;
9. $1 \to x = x$, $x \to 1 = 1$,
   $0 \to x = 1$, $x \to 0 = -x$;
10. $x \to y > W$ *iff* $x < W$ *or* $y > W$;
11. $x \to y < W$ *iff* $y < W$ *and* $x > W$;
12. $x \to y = 1$ *iff* $x = 0$ *or* $y = 1$.

In order to define the semantics of the hedge modification, we need to define some restrictions for HAs. We will use "hedge algebra" instead of "linear symmetric hedge algebra" in the rest of this paper.

2.2.2 Hedge Algebra as truth domain

Definition 14. *A HA* $AX = (X, G, H, >)$ *is* monotonic *if each* $h \in H^+(H^-)$ *is positive with respect to all* $k \in H^+(H^-)$*, and negative with respect to all* $k \in H^-(H^+)$*.*

We know that $H^+ \cup \{I\}$ and $H^- \cup \{I\}$ are linearly ordered, while $H \cup \{I\}$ is not. [5] extended the order relation on $H^+ \cup \{I\}$ and $H^- \cup \{I\}$ to one on $H \cup \{I\}$ as follows.

Definition 15. *Given* $h, k \in H \cup \{I\}$*,* $h \geq_h k$ *iff*

- $h \in H^+$ *and* $k \in H^-$*; or*
- $h, k \in H^+ \cup \{I\}$ *and* $h \geq k$*; or*
- $h, k \in H^- \cup \{I\}$ *and* $h \leq k$*.*

The following proposition which is presented in [5] shows that the hedge modifies the meaning of a linguistic value independently of preceding hedges in the hedge chain in monotonic HAs,

Proposition 16. *Consider a monotonic HA* $AX = (X, \{c^+, c^-\}, H, >)$*. Then*

$$h >_h k \Leftrightarrow h\sigma c^+ > k\sigma c^+$$

From Proposition 16 we have the following Corollary [5].

Corollary 17. *Consider a monotonic HA* $AX = (X, \{c^+, c^-\}, H, >)$*. Then*

1. $\forall h \in H^+, k \in H^-. h\sigma c^+ > \sigma c^+$ *and* $k\sigma c^+ < \sigma c^+$*.*
2. $h \geq k \Leftrightarrow h\sigma c^+ \geq k\sigma c^+$*.*

Similarly, the first hedge does not affect the meaning of the other.

Proposition 18. *Consider a monotonic HA* $AX = (X, \{c^+, c^-\}, H, >)$*. Then*

$$\forall h \in H.\ \sigma_1 c^+ > \sigma_2 c^+ \Leftrightarrow \sigma_1 h c^+ > \sigma_2 h c^+$$

The property holds when the generator is either c^+ or c^-.

Proposition 19. *Consider a monotonic HA* $AX = (X, \{c^+, c^-\}, H, >)$*. Then for* $c \in \{c^+, c^-\}$ *we have*

$$\forall h \in H.\ \sigma_1 c_1 > \sigma_2 c_2 \Leftrightarrow \sigma_1 h c_1 > \sigma_2 h c_2$$

The readers are referred to [5] for the proof of the last three propositions. A corollary is shown in [5] such that if a hedge is the first hedge or the last hedge in the hedge chain, then it is independent of other hedges and the generators.

Corollary 20. *Consider a monotonic HA* $AX = (X, \{c^+, c^-\}, H, >)$*. Then*

$$\forall \delta \in H^*.\ \sigma_1 c_1 > \sigma_2 c_2 \Leftrightarrow \sigma_1 \delta c_1 > \sigma_2 \delta c_2$$

2.2.3 Inverse mapping of hedges

Fuzzy description logics represent the assessment "It is true that Tom is very old" by

$$(VeryOld)^{\mathcal{I}}(Tom)^{\mathcal{I}} = True. \tag{2.1}$$

In a fuzzy linguistic logic [15, 16, 17], the assessment "It is true that Tom is very old" and the assessment "It is very true that Tom is old" are equivalent, which means

$$(Old)^{\mathcal{I}}(Tom)^{\mathcal{I}} = VeryTrue, \tag{2.2}$$

and (2.1) has the same meaning. This signifies that the modifier can be moved from concept term to truth value and vice versa. For any $h \in H$ and for any $\sigma \in H^*$, the rules of moving hedges [9] are as follows,

$$RT1 : (hC)^{\mathcal{I}}(d) = \sigma c \to (C)^{\mathcal{I}}(d) = \sigma hc$$
$$RT2 : (C)^{\mathcal{I}}(d) = \sigma hc \to (hC)^{\mathcal{I}}(d) = \sigma c.$$

where C is a concept term and $d \in \Delta^{\mathcal{I}}$.

Definition 21. *Consider a monotonic HA* $AX = (X, \{c^+, c^-\}, H, >)$ *and a* $h \in H$*. A mapping* $h^- : X \to X$ *is called an* inverse mapping *of* h *iff it satisfies the following two properties,*

1. $h^-(\sigma hc) = \sigma c$.
2. $\sigma_1 c_1 > \sigma_2 c_2 \Leftrightarrow h^-(\sigma_1 c_1) > h^-(\sigma_2 c_2)$.

where $c, c_1, c_2 \in G$*,* $h \in H$ *and* $\sigma_1, \sigma_2 \in H^*$*.*

Based on the definition of *inverse mapping of a hedge*, the rule $RT2$ has a generalized version,

$$GRT2 : (C)^{\mathcal{I}}(d) = \delta c \to (hC)^{\mathcal{I}}(d) = h^-(\delta c)$$

where $c, \in \{c^+, c^-\}$, $h \in H$ and $\delta \in H^*$, C is a concept term, h^- is inverse mapping of h.

The following Proposition [5] shows the general case of the Corollary 20.

Proposition 22. *Consider a monotonic HA* $AX = (X, \{c^+, c^-\}, H, >)$*. Then for* $c \in \{c^+, c^-\}$*, a hedge chain* δ *and its inverse mapping* δ^-*. then,*

$$\sigma_1 c_1 > \sigma_2 c_2 \text{ iff } \delta^-(\sigma_1 c_1) > \delta^-(\sigma_2 c_2)$$

2.3 $\mathcal{ALC}_{\mathcal{FL}}$

$\mathcal{ALC}_{\mathcal{FL}}$ is a Description Logic in which the truth domain of interpretations is represented by a hedge algebra. The syntax of $\mathcal{ALC}_{\mathcal{FL}}$ is similar to that of $\mathcal{ALCH}$ except that $\mathcal{ALC}_{\mathcal{FL}}$ allows concept modifiers and does not include role hierarchy.

Definition 23 (Syntax of $\mathcal{ALC}_{\mathcal{FL}}$ concept terms). *Let H be a set of hedges. Let A be a concept name and R a role, complex concept terms denoted by C, D in $\mathcal{ALC}_{\mathcal{FL}}$ are formed according to the following syntax rule:*

$$
\begin{array}{rcll}
C, D & \longrightarrow & A| & \textit{(concept name)} \\
 & & \top| & \textit{(top concept)} \\
 & & \bot| & \textit{(bottom concept)} \\
 & & C \sqcap D| & \textit{(concept conjunction)} \\
 & & C \sqcup D| & \textit{(concept disjunction)} \\
 & & \neg C| & \textit{(concept negation)} \\
 & & \delta C| & \textit{(modifier concept)} \\
 & & \forall R.C| & \textit{(universal quantification)} \\
 & & \exists R.C| & \textit{(existential quantification)}
\end{array}
$$

where $\delta \in H^*$.

The semantics is based on the notion of interpretations.

Definition 24 (Semantics of $\mathcal{ALC}_{\mathcal{FL}}$ concept terms). *Let AX be a monotonic HA such that $AX = (X, \{True, False\}, H, >)$. A* fuzzy interpretation ($\mathfrak{f}$-interpretation) *$\mathcal{I}$ for $\mathcal{ALC}_{\mathcal{FL}}$ is a pair $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a nonempty set and $\cdot^{\mathcal{I}}$ is an* interpretation function *mapping:*

- *individuals as for the classical case;*
- *a concept C into a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to X$;*
- *a role R into a function $R^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to X$.*

For all $d \in \Delta^{\mathcal{I}}$ the interpretation function satisfies the following equations

$$
\begin{array}{rcl}
\top^{\mathcal{I}}(d) & = & \sup(True), \\
\bot^{\mathcal{I}}(d) & = & \inf(False), \\
(\neg C)^{\mathcal{I}}(d) & = & -C^{\mathcal{I}}(d), \\
(C \sqcap D)^{\mathcal{I}}(d) & = & \min(C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)), \\
(C \sqcup D)^{\mathcal{I}}(d) & = & \max(C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)), \\
(\delta C)^{\mathcal{I}}(d) & = & \delta^{-}(C^{\mathcal{I}}(d)), \\
(\forall R.C)^{\mathcal{I}}(d) & = & \inf_{d' \in \Delta^{\mathcal{I}}}\{\max(-R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d'))\}, \\
(\exists R.C)^{\mathcal{I}}(d) & = & \sup_{d' \in \Delta^{\mathcal{I}}}\{\min(R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d'))\},
\end{array}
$$

where $-x$ is the contradictory element of x, and δ^{-} is the inverse of the hedge chain δ.

In order to insure the satisfiability preserving property, we consider only witnessed interpretations [2, 3]. For $\mathcal{ALC}_{\mathcal{FL}}$, an interpretation $\mathcal{I}$ is a *witnessed interpretation* if for all $d \in \Delta^{\mathcal{I}}$, for each concept term of the form $\forall R.C$ there exists a $d' \in \Delta^{\mathcal{I}}$ such that $(\forall R.C)^{\mathcal{I}}(d) = \max(-R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d'))$, and for each concept term of the form $\exists R.C$ there exists a $d' \in \Delta^{\mathcal{I}}$ such that $(\exists R.C)^{\mathcal{I}}(d) = \min(R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d'))$.

Definition 25 (Fuzzy Assertion). *A* fuzzy assertion (𝔣assertion) *is an expression of the form $\langle \alpha \bowtie \sigma c \rangle$ where α is of the form $a : C$ or $(a,b) : R$, $\bowtie \in \{\geq, >, \leq, <\}$ and $\sigma c \in X$.*

Formally, an 𝔣-interpretation $\mathcal{I}$ *satisfies* a fuzzy assertion $\langle a : C \geq \sigma c \rangle$ (respectively $\langle (a,b) : R \geq \sigma c \rangle$) iff $C^{\mathcal{I}}(a^{\mathcal{I}}) \geq \sigma c$ (respectively $R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \geq \sigma c$). An 𝔣-interpretation $\mathcal{I}$ *satisfies* a fuzzy assertion $\langle a : C \leq \sigma c \rangle$ (respectively $\langle (a,b) : R \leq \sigma c \rangle$) iff $C^{\mathcal{I}}(a^{\mathcal{I}}) \leq \sigma c$ (respectively $R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \leq \sigma c$). Similarly for $>$ and $<$.

Concerning terminological axioms, an $\mathcal{ALC}_{\mathcal{FL}}$ terminology axiom is of the form $C \sqsubseteq D$, where C and D are $\mathcal{ALC}_{\mathcal{FL}}$ concept terms. From a semantics point of view, a 𝔣-interpretation $\mathcal{I}$ *satisfies* a fuzzy concept inclusion $C \sqsubseteq D$ iff $\forall d \in \Delta^{\mathcal{I}}.C^{\mathcal{I}}(d) \leq D^{\mathcal{I}}(d)$. Two concept terms C, D are said to be *equivalent*, denoted by $C \equiv D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$ for all 𝔣-interpretations $\mathcal{I}$. Some properties concerning the hedge modification are showed in the following proposition [5].

Proposition 26. *We have the following semantical equivalence:*

$$\begin{array}{rcl} \delta(C \sqcap D) & \equiv & \delta(C) \sqcap \delta(D) \\ \delta(C \sqcup D) & \equiv & \delta(C) \sqcup \delta(D) \\ \delta_1(\delta_2 C) & \equiv & (\delta_1 \delta_2)C. \end{array}$$

According to Proposition 26, $Very(MoreC) = (VeryMore)C$ which is not the case in $\mathcal{ALC}_{\mathcal{FH}}$ [7] and $\mathcal{ALC}_{\mathcal{FLH}}$ [11].

Definition 27 (Fuzzy Knowledge Base). *A* fuzzy knowledge base (𝔣KB) *is a pair $\langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ and $\mathcal{A}$ are finite sets of terminological axioms and* 𝔣*assertions respectively.*

Example 28. *A 𝔣KB $\mathfrak{f}\mathcal{K} = \langle \{A \sqsubseteq \forall R.\neg B\}, \{a : \forall R.C \geq VeryTrue\} \rangle$.*

An 𝔣-interpretation $\mathcal{I}$ *satisfies* (is a *model* of) a TBox $\mathcal{T}$ iff $\mathcal{I}$ satisfies each element in $\mathcal{T}$. $\mathcal{I}$ *satisfies* (is a *model* of) an ABox $\mathcal{A}$ iff $\mathcal{I}$ satisfies each element in $\mathcal{A}$. $\mathcal{I}$ *satisfies* (is a *model* of) a 𝔣KB $\mathfrak{f}\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ iff $\mathcal{I}$ satisfies both $\mathcal{A}$ and $\mathcal{T}$. Given a 𝔣KB $\mathfrak{f}\mathcal{K}$ and a 𝔣assertion $\mathfrak{f}\alpha$. We say that $\mathfrak{f}\mathcal{K}$ *entails* $\mathfrak{f}\alpha$ (denoted $\mathfrak{f}\mathcal{K} \models \mathfrak{f}\alpha$) iff each model of $\mathfrak{f}\mathcal{K}$ satisfies $\mathfrak{f}\alpha$.

2.4 Examples of applications of hedges

In daily life, we often use hedges. We give three examples of applications of hedges to show why we need hedge-algebra OWL.

Example 29 (DNA testing). *DNA is the genetic material found within the cell nuclei of all living things. In mammals the strands of DNA are grouped into structures called chromosomes. With the exception of identical siblings (as in identical twins), the complete DNA of each individual is unique. DNA was first developed as an identification technique. Originally used to detect the presence of genetic diseases, it soon came to be used in criminal investigations and legal affairs. In criminal investigations, DNA derived from evidence collected at the crime scene are compared the DNA fingerprints of suspects. Generally, courts have accepted the reliability of DNA testing and admitted DNA testing results into evidence.*

DNA is also used to determine the filiation. Because there might be gene mutageneses and human error which could lead to false results, the DNA of the child might not match his parents' DNAs completely. Therefore, the DNA of child probably is a little different from his parents'. We can not say the fact of being the child of a certain parent is true only if their DNAs match 100%. Generally, courts accept the probability greater and equal to 90%. So it is more reasonable to say that if their DNAs are $VeryVeryVeryMatched$ *then "they are son and father" holds to a degree* $True$.

Hedges exist in our university as well.

Example 30. *Every professor has ever written recommendations for students and all students want their professors give them nice, veracious and just evaluations. If the professor defines some boundaries which divide marks into several classes such as "excellent" or "good", so that those who have marks better than 1.5 are excellent, and those who have marks between 1.5 and 2.5 are good and so forth, maybe the students who get marks of 1.6 are a little depressed because they are excluded from the excellent group though they are so close to it.*

By using hedges, that would be represented as the following. If John's mark is 1.0, the professor might think "John is a $Very$ *excellent student" holds to a degree* $VeryTrue$. *If John has mark of 1.6, that "John is an excellent student" holds to a degree* $True$ *would be appropriate. The following is an* $\mathcal{ALC}_{\mathcal{FL}}$ *knowledge base* $\mathcal{K}$ *of this situation.*

$$
\begin{aligned}
\mathcal{K} = \{ \quad & \langle John : \neg(\exists mark.VeryVeryGood) \sqcup Excellent \geq VeryTrue \rangle \\
& \langle (John, 1.0) : mark \geq VeryTrue \rangle \\
& \langle 1.0 : VeryVeryGood \geq VeryeryTrue \rangle \\
& \langle Mike : \neg(\exists mark.VeryVeryGood) \sqcup Excellent \geq True \rangle \\
& \langle (Mike, 1.6) : mark \geq VeryTrue \rangle \\
& \langle 1.6 : VeryGood \geq True \rangle \\
& \}
\end{aligned}
$$

Usually, when we talk about the weight we say it is heavy or light instead of the actual number. We can not fix a boundary such that a person is fat if his weight is heavier than the boundary otherwise he is not. We would rather say that it is $Probably$ true that somebody is $Very$ fat according to his weight. The following is an example of $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base. Tom wants to know whether he is fat. He builds the knowledge base and does the reasoning.

Example 31. *Consider a knowledge base $\mathcal{K}$:*

"A man is fat if it is very likely that he is very very heavy" holds to a degree at least True. For a man named Tom who want to know whether he is over normal weight:

$$\langle Tom : \neg(\exists weight.VeryVeryHeavy) \sqcup Fat \geq True\rangle \tag{2.3}$$

"The weight of Tom is 150kg " holds to a degree more than more-or-less True:

$$\langle (Tom, 150) : weight \geq MolTrue\rangle \tag{2.4}$$

"150kg is heavy " holds to a degree more than More True:

$$\langle 150 : Heavy \geq MoreTrue\rangle \tag{2.5}$$

Let us show that $\mathcal{K}$ entails that "Tom is fat" to a degree more than True. To prove this it suffices to prove $\mathcal{K}$ together with the following is unsatisfiable,

$$\langle Tom : Fat < True\rangle \tag{2.6}$$

We use the calculus introduced in [5] to solve this unsatisfiability problem in $\mathcal{ALC}_{\mathcal{FL}}$. The rules we use here are included in the calculus.

According the rule ($\sqcup_{\geq}$), we have two branches:

$$\langle Tom : \neg(\exists weight.VeryVeryHeavy) \geq True\rangle \tag{2.7}$$

$$\langle Tom : Fat \geq True\rangle \tag{2.8}$$

2.8 and 2.6 yield a clash. For 2.7, we apply the rule ($\neg_{\geq}$) and obtain:

$$\langle Tom : \exists weight.VeryVeryHeavy) \leq False\rangle \tag{2.9}$$

Since 2.4 is conjugated to $\langle (Tom, 150) : weight \leq False\rangle$, rule($\exists_{\leq}$) yields:

$$\langle 150 : VeryVeryHeavy \leq False\rangle \tag{2.10}$$

rule ($\delta_{\leq}$) yields:

$$\langle 150 : Heavy \leq VeryVeryFalse\rangle \tag{2.11}$$

2.11 has clash with 2.5.

There is no a clash free completion of $\mathcal{K} \cup \{\langle Tom : Fat < True\rangle\}$. Therefore $\mathcal{K} \cup \{\langle Tom : Fat < True\rangle\}$ is unsatisfiable. We can see that $\mathcal{K} \models \langle Tom : Fat \geq True\rangle$.

After the reasoning, Tom obtains the result which means he needs to lose weight.

To the best of our knowledge, there exists no implemented reasoners working with fuzzy DLs with hedges. Developing an optimized reasoner is a hard work and will cost a lot of time and material resources. It will be convenient if we can use existing reasoners to do reasoning in $\mathcal{ALC}_{\mathcal{FL}}$, so we transform $\mathcal{ALC}_{\mathcal{FL}}$ into classical DLs. In the next chapter, we will show such a transformation which is satisfiability preserving.

Chapter 3

Transforming $\mathcal{ALC}_{\mathcal{FL}}$ into $\mathcal{ALCH}$

3.1 The basic idea

We will introduce a satisfiability preserving transformation from $\mathcal{ALC}_{\mathcal{FL}}$ into $\mathcal{ALCH}$ in this section. First, we illustrate the basic idea the mapping relies on. The idea is similar to the one in [14].

Consider a monotonic HA $AX = (X, \{True, False\}, H, >)$. In the following, we assume that $c \in \{c^+, c^-\}$ where $c^+ = True, c^- = False$, $\sigma \in H^*$, $\sigma c \in X$ and $\bowtie \in \{\geq, >, \leq, <\}$. Assume we have an $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base, $\mathfrak{f}\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{A} = \{\mathfrak{f}\alpha_1, \mathfrak{f}\alpha_2, \mathfrak{f}\alpha_3, \mathfrak{f}\alpha_4\}$ and $\mathfrak{f}\alpha_1 = \langle a : A \geq True \rangle$, $\mathfrak{f}\alpha_2 = \langle b : A \geq VeryTrue \rangle$, $\mathfrak{f}\alpha_3 = \langle a : B \leq False \rangle$, and $\mathfrak{f}\alpha_4 = \langle b : B \leq VeryFalse \rangle$ where A, B are concept names. We introduce four new concept names: $A_{\geq True}$, $A_{\geq VeryTrue}$, $B_{\leq False}$ and $B_{\leq VeryFalse}$. The concept name $A_{\geq True}$ represents the set of individuals that are instances of A with degree greater and equal to $True$. The concept name $B_{\leq VeryFlase}$ represents the set of individuals that are instances of B with degree less and equal to $VeryFalse$. We can map the fuzzy assertions into classical assertions:

$$\langle a : A \geq True \rangle \rightarrow \langle a : A_{\geq True} \rangle,$$

$$\langle b : A \geq VeryTrue \rangle \rightarrow \langle b : A_{\geq VeryTrue} \rangle,$$

$$\langle a : B \leq False \rangle \rightarrow \langle a : B_{\leq False} \rangle,$$

$$\langle b : B \leq VeryFalse \rangle \rightarrow \langle b : B_{\leq VeryFalse} \rangle.$$

We also need to consider the relationships among the newly introduced concept names. Because $VeryTrue > True$, it is easy to get if a truth value $\sigma c \geq VeryTrue$ then $\sigma c \geq True$. Thus, we obtain a new inclusion $A_{\geq VeryTrue} \sqsubseteq A_{\geq True}$. Similarly for B, because $VeryFalse < False$, a truth value $\sigma c \leq VeryFalse$ implies $\sigma c \leq False$ too. Then the inclusion $B_{\leq VeryFalse} \sqsubseteq B_{\leq False}$ is obtained.

Now, let us proceed with the mappings. Let $\mathfrak{f}\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be an $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base. We are going to transform $\mathfrak{f}\mathcal{K}$ into an $\mathcal{ALCH}$ knowledge base $\mathcal{K}$. We assume $\sigma c \in [\inf(False), \sup(True)]$ and $\bowtie \in \{\geq, >, \leq, <\}$.

3.2 The transformation of ABox

In order to transform $\mathcal{A}$, we define two mappings θ and ρ to map all the assertions in $\mathcal{A}$ into classical assertions. Notice that we do not allow assertions of the forms $(a.b) : R < \sigma c$ and $(a.b) : R \leq \sigma c$ because they related to 'negated role' which is not part of classical $\mathcal{ALCH}$. Most of reasoners do not support $\mathcal{ALCH}(\neg)$ which is $\mathcal{ALCH}$ with negated role and the complexity of $\mathcal{ALCH}(\neg)$ is still open.

3.2.1 The mapping θ

θ maps fuzzy assertions into classical assertions using ρ. We define it as follows.

Definition 32. *Let $\mathfrak{f}\alpha$ be a $\mathfrak{f}$assertion in $\mathcal{A}$, then,*

$$\theta(\mathfrak{f}\alpha) = \begin{cases} a : \rho(C, \bowtie \sigma c) & \text{if } \ \mathfrak{f}\alpha = \langle a : C \bowtie \sigma c \rangle \\ (a, b) : \rho(R, \bowtie \sigma c) & \text{if } \ \mathfrak{f}\alpha = \langle (a, b) : R \bowtie \sigma c \rangle. \end{cases}$$

We extend θ to a set of $\mathfrak{f}$assertions $\mathcal{A}$ point-wise,

$$\theta(\mathcal{A}) = \{\theta(\mathfrak{f}\alpha) \mid \mathfrak{f}\alpha \in \mathcal{A}\}.$$

Let's see how the mapping ρ is defined next.

3.2.2 The mapping ρ

The mapping ρ combines the $\mathcal{ALC}_{\mathcal{FL}}$ concept term, the $\bowtie$ and the fuzzy value σc together into an $\mathcal{ALCH}$ concept term.

Let A be a concept name, C, D be concept terms and R be a role name. For roles we have simply

$$\rho(R, \bowtie \sigma c) = R_{\bowtie \sigma c}.$$

Example 33. *Let $\mathfrak{f}\alpha$ be a role assertion and $\mathfrak{f}\alpha = \langle (a, b) : R \geq VeryTrue \rangle$, then $\theta(\mathfrak{f}\alpha) = (a, b) : \rho(R, \geq VeryTrue) = (a, b) : R_{\geq VeryTrue}$.*

For concept terms, the mapping ρ is inductively defined on the structures of concept terms:
For $\top$,

$$\rho(\top, \bowtie \sigma c) = \begin{cases} \top & \text{if} \quad \bowtie \sigma c = \ \geq \sigma c \\ \top & \text{if} \quad \bowtie \sigma c = \ > \sigma c, \sigma c < \sup(c^+) \\ \bot & \text{if} \quad \bowtie \sigma c = \ > \sup(c^+) \\ \top & \text{if} \quad \bowtie \sigma c = \ \leq \sup(c^+) \\ \bot & \text{if} \quad \bowtie \sigma c = \ \leq \sigma c, \sigma c < \sup(c^+) \\ \bot & \text{if} \quad \bowtie \sigma c = \ < \sigma c. \end{cases}$$

For $\bot$,

$$\rho(\bot, \bowtie \sigma c) = \begin{cases} \top & \text{if} \quad \bowtie \sigma c = \quad \geq \inf(c^-) \\ \bot & \text{if} \quad \bowtie \sigma c = \quad \geq \sigma c, \sigma c > \inf(c^-) \\ \bot & \text{if} \quad \bowtie \sigma c = \quad > \sigma c \\ \top & \text{if} \quad \bowtie \sigma c = \quad \leq \sigma c \\ \top & \text{if} \quad \bowtie \sigma c = \quad < \sigma c, \sigma c > \inf(c^-) \\ \bot & \text{if} \quad \bowtie \sigma c = \quad < \inf(c^-). \end{cases}$$

For concept name A,

$$\rho(A, \bowtie \sigma c) = A_{\bowtie \sigma c}.$$

For concept conjunction $C \sqcap D$,

$$\rho(C \sqcap D, \bowtie \sigma c) = \begin{cases} \rho(C, \bowtie \sigma c) \sqcap \rho(D, \bowtie \sigma c) & \text{if} \quad \bowtie \in \{\geq, >\} \\ \rho(C, \bowtie \sigma c) \sqcup \rho(D, \bowtie \sigma c) & \text{if} \quad \bowtie \in \{\leq, <\}. \end{cases}$$

Example 34. *Let* $\mathfrak{f}\alpha = \langle a : A \sqcap B \geq VeryTrue \rangle$, *then*

$$\begin{aligned} \theta(\mathfrak{f}\alpha) &= a : \rho(A \sqcap B, \geq VeryTrue) \\ &= a : \rho(A, \geq VeryTrue) \sqcap \rho(B, \geq VeryTrue) \\ &= a : A_{\geq VeryTrue} \sqcap B_{\geq VeryTrue}. \end{aligned}$$

For concept disjunction $C \sqcup D$,

$$\rho(C \sqcup D, \bowtie \sigma c) = \begin{cases} \rho(C, \bowtie \sigma c) \sqcup \rho(D, \bowtie \sigma c) & \text{if} \quad \bowtie \in \{\geq, >\} \\ \rho(C, \bowtie \sigma c) \sqcap \rho(D, \bowtie \sigma c) & \text{if} \quad \bowtie \in \{\leq, <\}. \end{cases}$$

Example 35. *Let* $\mathfrak{f}\alpha = \langle a : A \sqcup B \leq VeryTrue \rangle$, *then*

$$\begin{aligned} \theta(\mathfrak{f}\alpha) &= a : \rho(A \sqcup B, \leq VeryTrue) \\ &= a : \rho(A, \leq VeryTrue) \sqcap \rho(B, \leq VeryTrue) \\ &= a : A_{\leq VeryTrue} \sqcap B_{\leq VeryTrue}. \end{aligned}$$

For concept negation $\neg C$

$$\rho(\neg C, \bowtie \sigma c) = \rho(C, \neg \bowtie \sigma \bar{c}),$$

where $\neg \geq\ = \ \leq, \neg >\ =\ <, \neg \leq\ =\ \geq, \neg <\ =\ >$ and

$$\bar{c} = \begin{cases} c^+ & \text{if} \quad c = c^- \\ c^- & \text{if} \quad c = c^+. \end{cases}$$

Example 36. *Let* $\mathfrak{f}\alpha = \langle a : \neg A \geq LessTrue \rangle$, *then*

$$\theta(\mathfrak{f}\alpha) = a : \rho(\neg A, \geq LessTrue) = a : \rho(A, \leq LessFalse) = a : A_{\leq LessFalse}.$$

For modifier concept δC,

$$\rho(\delta C, \bowtie \sigma c) = \rho(C, \bowtie \sigma\delta c).$$

Example 37. *Let* $\mathfrak{f}\alpha = \langle a : Very(A \sqcap B) \leq LessFalse\rangle$, *then*

$$\begin{aligned}\theta(\mathfrak{f}\alpha) &= a : \rho(Very(A \sqcap B), \leq LessFalse)\\ &= a : \rho((A \sqcap B), \leq LessVeryFalse)\\ &= a : \rho(A, \leq LessVeryFalse) \sqcup \rho(B, \leq LessVeryFalse)\\ &= a : A_{\leq LessVeryTrue} \sqcup B_{\leq LessVeryTrue}.\end{aligned}$$

For existential quantification $\exists R.C$,

$$\rho(\exists R.C, \bowtie \sigma c) = \begin{cases} \exists\rho(R, \bowtie \sigma c).\rho(C, \bowtie \sigma c) & \text{if } \bowtie \in \{\geq, >\}\\ \forall\rho(R, - \bowtie \sigma c).\rho(C, \bowtie \sigma c) & \text{if } \bowtie \in \{\leq, <\},\end{cases}$$

where $- \leq = >$ and $- < = \geq$.

Example 38. *Let* $\mathfrak{f}\alpha = \langle a : \exists R.A \leq VeryTrue\rangle$, *then*

$$\begin{aligned}\theta(\mathfrak{f}\alpha) &= a : \rho(\exists R.A, \leq VeryTrue)\\ &= a : \forall\rho(R, > VeryTrue).\rho(A, \leq VeryTrue)\\ &= a : \forall R_{>VeryTrue}.\rho(A, \leq VeryTrue)\\ &= a : \forall R_{>VeryTrue}.A_{\leq VeryTrue}.\end{aligned}$$

For universal quantification $\forall R.C$,

$$\rho(\forall R.C, \bowtie \sigma c) = \begin{cases} \forall\rho(R, + \bowtie \sigma\bar{c}).\rho(C, \bowtie \sigma c) & \text{if } \bowtie \in \{\geq, >\}\\ \exists\rho(R, \neg \bowtie \sigma\bar{c}).\rho(C, \bowtie \sigma c) & \text{if } \bowtie \in \{\leq, <\},\end{cases}$$

where $+ \geq = >$ and $+ > = \geq$.

Example 39. *Let* $\mathfrak{f}\alpha = \langle a : \forall R.B > VeryTrue\rangle$, *then*

$$\begin{aligned}\theta(\mathfrak{f}\alpha) &= a : \rho(\forall R.B, > VeryTrue)\\ &= a : \forall\rho(R, \geq VeryFalse).\rho(B, > VeryTrue)\\ &= a : \forall R_{\geq VeryFalse}.\rho(B, > VeryTrue)\\ &= a : \forall R_{\geq VeryFalse}.B_{>VeryTrue}.\end{aligned}$$

According to the rules above, we can see that $|\theta(\mathcal{A})|$ is linearly bounded by $|\mathcal{A}|$.

3.3 The transformation of TBox

The new TBox is a union of two terminologies. One is the newly introduced TBox (denoted by $\mathcal{T}(N^{\mathfrak{f}\mathcal{K}})$ which is the terminology relating to the newly introduced concept names and role names. The other one is $\kappa(\mathfrak{f}\mathcal{K}, \mathcal{T})$ which is reduced by a mapping κ from the TBox of an $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base.

3.3.1 The newly introduced TBox

Many new concept names and role names are introduced when we transform an ABox. We need a set of terminological axioms to define the relationships among those new names.

We collect all the linguist terms σc that might be the subscript of a concept name or a role name. Let A be a concept name, R be a role name.

$$\begin{aligned} X^{\mathfrak{f}\mathcal{K}} = \; & \{\sigma c \mid \langle \alpha \bowtie \sigma c\rangle \in \mathcal{A}\} \\ & \cup \{\sigma c \mid \rho(A, \bowtie \sigma c)\} \\ & \cup \{\sigma c \mid \rho(R, \bowtie \sigma c)\}. \end{aligned}$$

We define a sorted set of linguistic terms,

$$\begin{aligned} N^{\mathfrak{f}\mathcal{K}} &= \{\inf(False), W, \sup(True)\} \\ & \quad \cup X^{\mathfrak{f}\mathcal{K}} \cup \{\sigma\bar{c} \mid \sigma c \in X^{\mathfrak{f}\mathcal{K}}\} \\ &= \{n_1, \ldots, n_{|N^{\mathfrak{f}\mathcal{K}}|}\} \end{aligned}$$

where $W = \inf(True) = \sup(False), n_i < n_{i+1}$ for $1 \le i \le |N^{\mathfrak{f}K}| - 1$ and $n_1 = \inf(False), n_{|N^{\mathfrak{f}K}|} = \sup(True)$.

Example 40. *Consider Example 28, the sorted set is,*

$$N^{\mathfrak{f}\mathcal{K}} = \{\inf(False), VeryFalse, W, VeryTrue, \sup(True)\}.$$

Let $\mathcal{T}(N^{\mathfrak{f}\mathcal{K}})$ be the set of terminological axioms relating to the newly introduced concept names and role names.

Definition 41 ($\mathcal{T}(N^{\mathfrak{f}\mathcal{K}})$). *Let $\mathcal{A}^{\mathfrak{f}\mathcal{K}}$ and $\mathcal{R}^{\mathfrak{f}\mathcal{K}}$ be the sets of concept names and role names occurring in $\mathfrak{f}\mathcal{K}$ respectively. For each $A \in \mathcal{A}^{\mathfrak{f}\mathcal{K}}$, for each $R \in \mathcal{R}^{\mathfrak{f}\mathcal{K}}$, for each $1 \le i \le |N^{\mathfrak{f}K}| - 1$ and for each $2 \le j \le |N^{\mathfrak{f}K}|$, $\mathcal{T}(N^{\mathfrak{f}\mathcal{K}})$ contains*

$$\begin{aligned} A_{\ge n_{i+1}} &\sqsubseteq A_{>n_i}, \\ A_{>n_i} &\sqsubseteq A_{\ge n_i}, \\ A_{<n_j} &\sqsubseteq A_{\le n_j}, \\ A_{\le n_i} &\sqsubseteq A_{<n_{i+1}}, \\ A_{\ge n_j} \sqcap A_{<n_j} &\sqsubseteq \bot, \\ A_{>n_i} \sqcap A_{\le n_i} &\sqsubseteq \bot, \\ \top &\sqsubseteq A_{\ge n_j} \sqcup A_{<n_j}, \\ \top &\sqsubseteq A_{>n_i} \sqcup A_{\le n_i}, \\ R_{\ge n_{i+1}} &\sqsubseteq R_{>n_i}, \\ R_{>n_i} &\sqsubseteq R_{\ge n_i}. \end{aligned}$$

where $n \in N^{\mathfrak{f}K}$.

$n_{i+1} > n_i$ because $N^{\mathfrak{f}\mathcal{K}}$ is a sorted set. Then if an individual is an instance of a concept name with degree $\ge n_{i+1}$ then the degree is also $> n_i$. The first terminological axiom shows that if an individual is an instance of $A_{\geq n_{i+1}}$ then it is an instance of $A_{>n_i}$ as well. Similarly, if an individual is an instance of a concept name with degree $\leq n_i$ then the degree is also $< n_{i+1}$. The third terminological axiom shows that if an individual is an instance of $A_{\leq n_i}$ then it is also an instance of $A_{<n_{i+1}}$. $A_{\geq n_j} \sqcap A_{<n_j} \sqsubseteq \bot$ because there is no individual such that it is an instance of a concept name with degree $\geq n_j$ and with degree $< n_j$ at the same time.

$\mathcal{T}(N^{\mathfrak{f}\mathcal{K}})$ contains $8|\mathcal{A}^{\mathfrak{f}\mathcal{K}}|(|\mathcal{N}^{\mathfrak{f}\mathcal{K}}|-1)$ plus $2|\mathcal{R}^{\mathfrak{f}\mathcal{K}}|(|\mathcal{N}^{\mathfrak{f}\mathcal{K}}|-1)$ terminological axioms.

Example 42. *Consider the $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base in Example 28, the following is an excerpt of the $\mathcal{T}(N^{\mathfrak{f}\mathcal{K}})$,*

$$\begin{array}{rcl} \mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) & = & \{A_{\geq \sup(True)} \sqsubseteq A_{>VeryTrue}, A_{\geq VeryTrue} \sqsubseteq A_{>W}, \\ & & A_{\geq W} \sqsubseteq A_{>VeryFalse}, A_{\geq VeryFalse} \sqsubseteq A_{>\inf(False)}\} \\ & & \cup \{\ldots, A_{\geq VeryTrue} \sqcap A_{<VeryTrue} \sqsubseteq \bot, \ldots\} \\ & & \cup \{\ldots, \top \sqsubseteq A_{\geq VeryTrue} \sqcup A_{<VeryTrue}, \ldots\} \\ & & \cup \{\ldots, R_{\geq \sup(True)} \sqsubseteq R_{>VeryTrue}, \ldots\}. \end{array}$$

3.3.2 The mapping κ

κ maps the fuzzy TBox into classical TBox.

Definition 43 **($\kappa(\mathfrak{f}\mathcal{K}, \mathcal{T})$).** *Let C, D be two concept terms and $C \sqsubseteq D \in \mathcal{T}$. For all $n \in N^{\mathfrak{f}\mathcal{K}}$*

$$\begin{array}{rl} \kappa(\mathfrak{f}\mathcal{K}, C \sqsubseteq D) = & \bigcup_{n \in N^{\mathfrak{f}\mathcal{K}}, \bowtie \in \{\geq, >\}} \{\rho(C, \bowtie n) \sqsubseteq \rho(D, \bowtie n)\} \\ & \bigcup_{n \in N^{\mathfrak{f}\mathcal{K}}, \bowtie \in \{\leq, <\}} \{\rho(D, \bowtie n) \sqsubseteq \rho(C, \bowtie n)\} \end{array} \tag{3.1}$$

We extend κ to a terminology $\mathcal{T}$ point-wise. For all $\tau \in \mathcal{T}$

$$\kappa(\mathfrak{f}\mathcal{K}, \mathcal{T}) = \cup_{\tau \in \mathcal{T}} \kappa(\mathfrak{f}\mathcal{K}, \tau).$$

κ reduces a terminological axiom in $\mathcal{ALC}_{\mathcal{FL}}$ into a set of $\mathcal{ALCH}$ terminology axioms.

3.4 The satisfiability preserving theorem

We now have all the ingredients to complete the reduction of an $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base into an $\mathcal{ALCH}$ knowledge base. The *reduction* of $\mathfrak{f}\mathcal{K}$ into an $\mathcal{ALCH}$ knowledge base, denoted $\mathcal{K}(\mathfrak{f}\mathcal{K})$, is defined as

$$\mathcal{K}(\mathfrak{f}\mathcal{K}) = \langle \mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) \cup \kappa(\mathfrak{f}\mathcal{K}, \mathcal{T}), \theta(\mathcal{A}) \rangle.$$

The soundness and completeness of the algorithm can be guaranteed by the following satisfiability preserving reduction theorem.

Theorem 44. *Let $\mathfrak{f}\mathcal{K}$ be an $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base. Then $\mathfrak{f}\mathcal{K}$ is satisfiable iff the $\mathcal{ALCH}$ knowledge base $\mathcal{K}(\mathfrak{f}\mathcal{K})$ is satisfiable.*

Proof. Let $\mathfrak{f}\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be an $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base , $\mathcal{K}(\mathfrak{f}\mathcal{K}) = \langle \mathcal{T}', \mathcal{A}' \rangle$ be the transformed $\mathcal{ALCH}$ knowledge base, where $\mathcal{T}' = \mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) \cup \kappa(\mathfrak{f}\mathcal{K}, \mathcal{T})$ and $\mathcal{A}' = \theta(\mathcal{A})$. We define that $\rhd \in \{\geq, >\}$ and $\lhd \in \{\leq, <\}$.

Our goal is to prove $\exists \mathcal{I}.\mathcal{I} \models \mathfrak{f}\mathcal{K} \Leftrightarrow \exists \mathcal{I}'.\mathcal{I}' \models \mathcal{K}(\mathfrak{f}\mathcal{K})$, where $\mathcal{I}$ is a fuzzy interpretation and $\mathcal{I}'$ is an $\mathcal{ALCH}$ interpretation.

$\Rightarrow$.) Assume $\mathcal{I}$ is a witnessed interpretation, such that $\mathcal{I} \models \mathfrak{f}\mathcal{K}$. We construct an $\mathcal{ALCH}$ interpretation $\mathcal{I}'$:

- $\Delta^{\mathcal{I}'} := \Delta^{\mathcal{I}}$,
- $a^{\mathcal{I}'} := a^{\mathcal{I}}$ for all individual a,
- $A^{\mathcal{I}'}_{\bowtie \sigma c} := \{d \in \Delta^{\mathcal{I}'} \mid A^{\mathcal{I}}(d) \bowtie \sigma c\}$, for all concept name $A_{\bowtie \sigma c}$,
- $R^{\mathcal{I}'}_{\bowtie \sigma c} := \{(d, d') \in \Delta^{\mathcal{I}'} \times \Delta^{\mathcal{I}'} \mid R^{\mathcal{I}}(d, d') \bowtie \sigma c\}$, for all role name $R_{\bowtie \sigma c}$.

(1) First, let's prove the follow Lemma.

Lemma 45. *Let C be a concept term in $\mathcal{ALC}_{\mathcal{FL}}$. $C \neq \top$ and $C \neq \bot$. It follows that $(\rho(C, \bowtie \sigma c))^{\mathcal{I}'} = \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \bowtie \sigma c\}$.*

Proof. Let R be a role name,
$(\rho(R, \bowtie \sigma c))^{\mathcal{I}'} = R^{\mathcal{I}'}_{\bowtie \sigma c} = \{(d, d') \in \Delta^{\mathcal{I}'} \times \Delta^{\mathcal{I}'} \mid R^{\mathcal{I}}(d, d') \bowtie \sigma c\}$.
We prove inductively on the structures of concept terms.
Let A be a concept name,
$(\rho(A, \bowtie \sigma c))^{\mathcal{I}'} = A^{\mathcal{I}'}_{\bowtie \sigma c} = \{d \in \Delta^{\mathcal{I}'} \mid A^{\mathcal{I}}(d) \bowtie \sigma c\}$.
Let C be a concept term. Assume

$$(\rho(C, \bowtie \sigma c))^{\mathcal{I}'} = \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \bowtie \sigma c\}.$$

For $\neg C$,

$$\begin{aligned} (\rho(\neg C, \bowtie \sigma c))^{\mathcal{I}'} &= (\rho(C, \neg \bowtie \sigma \bar{c}))^{\mathcal{I}'} \\ &\overset{\text{I.H.}}{=} \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \neg \bowtie \sigma \bar{c}\} \\ &= \{d \in \Delta^{\mathcal{I}'} \mid (\neg C)^{\mathcal{I}}(d) \bowtie \sigma c\}. \end{aligned}$$

For δC,

$$\begin{aligned} (\rho(\delta C, \bowtie \sigma c))^{\mathcal{I}'} &= (\rho(C, \bowtie \sigma \delta c))^{\mathcal{I}'} \\ &\overset{\text{I.H.}}{=} \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \bowtie \sigma \delta c\} \\ &= \{d \in \Delta^{\mathcal{I}'} \mid (\delta C)^{\mathcal{I}}(d) \bowtie \sigma c\}. \end{aligned}$$

For $C \sqcap D$,

$$\begin{aligned} (\rho(C \sqcap D, \rhd \sigma c))^{\mathcal{I}'} &= (\rho(C, \rhd \sigma c) \sqcap \rho(D, \rhd \sigma c))^{\mathcal{I}'} \\ &\overset{\text{I.H.}}{=} \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \rhd \sigma c\} \cap \{d \in \Delta^{\mathcal{I}'} \mid D^{\mathcal{I}}(d) \rhd \sigma c\}. \\ &= \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \rhd \sigma c \wedge D^{\mathcal{I}}(d) \rhd \sigma c\}. \\ &= \{d \in \Delta^{\mathcal{I}'} \mid \min(C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)) \rhd \sigma c\}. \\ &= \{d \in \Delta^{\mathcal{I}'} \mid (C \sqcap D)^{\mathcal{I}}(d) \rhd \sigma c\}. \end{aligned}$$

$$\begin{array}{rcl}
(\rho(C \sqcap D, \lhd \sigma c))^{\mathcal{I}'} & = & (\rho(C, \lhd \sigma c) \sqcup \rho(D, \lhd \sigma c))^{\mathcal{I}'} \\
& \overset{\text{I.H.}}{=} & \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \lhd \sigma c \ \} \cup \{d \in \Delta^{\mathcal{I}'} \mid D^{\mathcal{I}}(d) \lhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \lhd \sigma c \vee D^{\mathcal{I}}(d) \lhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \min(C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)) \lhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid (C \sqcap D)^{\mathcal{I}}(d) \lhd \sigma c\}.
\end{array}$$

For $C \sqcup D$,

$$\begin{array}{rcl}
(\rho(C \sqcup D, \rhd \sigma c))^{\mathcal{I}'} & = & (\rho(C, \rhd \sigma c) \sqcup \rho(D, \rhd \sigma c))^{\mathcal{I}'} \\
& \overset{\text{I.H.}}{=} & \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \rhd \sigma c \ \} \cup \{d \in \Delta^{\mathcal{I}'} \mid D^{\mathcal{I}}(d) \rhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \rhd \sigma c \vee D^{\mathcal{I}}(d) \rhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \max(C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)) \rhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid (C \sqcup D)^{\mathcal{I}}(d) \rhd \sigma c\}.
\end{array}$$

$$\begin{array}{rcl}
(\rho(C \sqcup D, \lhd \sigma c))^{\mathcal{I}'} & = & (\rho(C, \lhd \sigma c) \sqcap \rho(D, \lhd \sigma c))^{\mathcal{I}'} \\
& \overset{\text{I.H.}}{=} & \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \lhd \sigma c \ \} \cap \{d \in \Delta^{\mathcal{I}'} \mid D^{\mathcal{I}}(d) \lhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d) \lhd \sigma c \wedge D^{\mathcal{I}}(d) \lhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \max(C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)) \lhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid (C \sqcup D)^{\mathcal{I}}(d) \lhd \sigma c\}.
\end{array}$$

For $\forall R.C$,

$$\begin{array}{rcl}
(\rho(\forall R.C, \rhd \sigma c))^{\mathcal{I}'} & = & (\forall \rho(R, + \rhd \sigma \bar{c}).\rho(C, \rhd \sigma c))^{\mathcal{I}'} \\
& \overset{\text{I.H.}}{=} & \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(d, d') \notin R^{\mathcal{I}'}_{+\rhd\sigma\bar{c}} \vee C^{\mathcal{I}}(d') \rhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(d, d') \in R^{\mathcal{I}'}_{\neg\rhd\sigma\bar{c}} \vee C^{\mathcal{I}}(d') \rhd \sigma c\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \bigwedge_{d' \in \Delta^{\mathcal{I}'}} (R^{\mathcal{I}}(d, d') \neg \rhd \sigma \bar{c} \vee C^{\mathcal{I}}(d') \rhd \sigma c)\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \bigwedge_{d' \in \Delta^{\mathcal{I}'}} (-R^{\mathcal{I}}(d, d') \rhd \sigma c \vee C^{\mathcal{I}}(d') \rhd \sigma c)\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \bigwedge_{d' \in \Delta^{\mathcal{I}'}} (\max(-R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')) \rhd \sigma c)\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \inf_{d' \in \Delta^{\mathcal{I}'}} \ (\max(-R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')\} \rhd \sigma c)\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid (\forall R.C)^{\mathcal{I}}(d) \rhd \sigma c)\}.
\end{array}$$

$$\begin{array}{rcl}
(\rho(\forall R.C, \lhd \sigma c))^{\mathcal{I}'} & = & (\exists \rho(R, \neg \lhd \sigma \bar{c}).\rho(C, \lhd \sigma c))^{\mathcal{I}'} \\
& \overset{\text{I.H.}}{=} & \{d \in \Delta^{\mathcal{I}'} \mid \exists d' \in \Delta^{\mathcal{I}'}.(d, d') \in R^{\mathcal{I}'}_{\neg\lhd\sigma\bar{c}} \wedge C^{\mathcal{I}}(d') \lhd \sigma c)\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \bigvee_{d' \in \Delta^{\mathcal{I}'}} (R^{\mathcal{I}}(d, d') \neg \lhd \sigma \bar{c} \wedge C^{\mathcal{I}}(d') \lhd \sigma c)\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \bigvee_{d' \in \Delta^{\mathcal{I}'}} (-R^{\mathcal{I}}(d, d') \lhd \sigma c \wedge C^{\mathcal{I}}(d') \lhd \sigma c)\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \bigvee_{d' \in \Delta^{\mathcal{I}'}} (\max(-R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')) \lhd \sigma c)\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid \inf_{d' \in \Delta^{\mathcal{I}'}} \ (\max(-R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')\} \lhd \sigma c)\} \\
& = & \{d \in \Delta^{\mathcal{I}'} \mid (\forall R.C)^{\mathcal{I}}(d) \lhd \sigma c)\}.
\end{array}$$

For $\exists R.C$,

$$\begin{array}{rcl}(\rho(\exists R.C, \rhd \sigma c))^{\mathcal{I}'} & = & (\exists \rho(R, \rhd \sigma c).\rho(C, \rhd \sigma c))^{\mathcal{I}'} \\ & \overset{\text{I.H.}}{=} & \{d \in \Delta^{\mathcal{I}'} \mid \exists d' \in \Delta^{\mathcal{I}'}.(d, d') \in R^{\mathcal{I}'}_{\rhd \sigma c} \wedge C^{\mathcal{I}}(d') \rhd \sigma c)\} \\ & = & \{d \in \Delta^{\mathcal{I}'} \mid \bigvee_{d' \in \Delta^{\mathcal{I}'}} (R^{\mathcal{I}'}(d, d') \rhd \sigma c \wedge C^{\mathcal{I}}(d') \rhd \sigma c)\} \\ & = & \{d \in \Delta^{\mathcal{I}'} \mid \bigvee_{d' \in \Delta^{\mathcal{I}'}} (\min(R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')) \rhd \sigma c)\} \\ & = & \{d \in \Delta^{\mathcal{I}'} \mid \sup_{d' \in \Delta^{\mathcal{I}'}} \{\min(R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')\} \rhd \sigma c)\} \\ & = & \{d \in \Delta^{\mathcal{I}'} \mid (\exists R.C)^{\mathcal{I}}(d) \rhd \sigma c)\}.\end{array}$$

$$\begin{array}{rcl}(\rho(\exists R.C, \lhd \sigma c))^{\mathcal{I}'} & = & (\forall \rho(R, - \lhd \sigma c).\rho(C, \lhd \sigma c))^{\mathcal{I}'} \\ & \overset{\text{I.H.}}{=} & \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(d, d') \notin R^{\mathcal{I}'}_{- \lhd \sigma c} \vee C^{\mathcal{I}}(d') \lhd \sigma c)\} \\ & = & \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(d, d') \in R^{\mathcal{I}'}_{\lhd \sigma c} \vee C^{\mathcal{I}}(d') \lhd \sigma c)\} \\ & = & \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(R^{\mathcal{I}}(d, d') \lhd \sigma c \vee C^{\mathcal{I}}(d') \lhd \sigma c)\} \\ & = & \{d \in \Delta^{\mathcal{I}'} \mid \bigwedge_{d' \in \Delta^{\mathcal{I}'}} (\min(R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')) \lhd \sigma c)\} \\ & = & \{d \in \Delta^{\mathcal{I}'} \mid \sup_{d' \in \Delta^{\mathcal{I}'}} (\min(R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')\} \lhd \sigma c)\} \\ & = & \{d \in \Delta^{\mathcal{I}'} \mid (\exists R.C)^{\mathcal{I}}(d) \lhd \sigma c)\}.\end{array}$$

In the following, we use $C_{\bowtie \sigma c}$ to represent $\rho(C, \bowtie \sigma c)$. □

(2) Now we prove that for all $\alpha \bowtie \sigma c \in \mathcal{A}$,

$$\mathcal{I} \models \alpha \bowtie \sigma c \Rightarrow \mathcal{I}' \models \theta(\alpha \bowtie \sigma c).$$

For role assertion,

$$\begin{array}{rcl}\mathcal{I} \models (a, b) : R \bowtie \sigma c & \Rightarrow & R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \bowtie \sigma c \\ & \Rightarrow & (a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in R^{\mathcal{I}'}_{\bowtie \sigma c} \\ & \Rightarrow & \mathcal{I}' \models (a, b) : R_{\bowtie \sigma c}.\end{array}$$

For concept assertions, we inductively prove on the structure of concept term:
For $\top$,
For all interpretation $\mathcal{I}$ and for all $d \in \Delta^{\mathcal{I}}$, $\top^{\mathcal{I}}(d) = \sup(True)$, so $a : \top \geq \sigma c, a : \top > \sigma c$ if $\sigma c < \sup(True)$ and $a : \top \leq \sup(True)$ are valid, $a : \top$ is valid too. While $a : \top > \sup(True), a : \top \leq \sigma c$ if $\sigma c < \sup(True)$ and $a : \top < \sigma c$ are unsatisfiable, $a : \bot$ is unsatisfiable as well.
For $\bot$,
For all interpretation $\mathcal{I}$ and for all $d \in \Delta^{\mathcal{I}}$, $\bot^{\mathcal{I}}(d) = \inf(False)$, so $a : \bot \geq \inf(False), a : \bot < \sigma c$ if $\sigma c > \inf(False)$ and $a : \bot \leq \sigma c$ are valid, so is $a : \top$. While $a : \bot < \inf(False), a : \bot \geq \sigma c$ if $\sigma c > \inf(False)$ and $a : \bot > \sigma c$ are unsatisfiable. $a : \bot$ is also unsatisfiable.
For concept name A,
$\mathcal{I} \models a : A \bowtie \sigma c \Rightarrow A^{\mathcal{I}}(a^{\mathcal{I}}) \bowtie \sigma c \Rightarrow a^{\mathcal{I}'} \in A^{\mathcal{I}'}_{\bowtie \sigma c} \Rightarrow \mathcal{I}' \models a : A_{\bowtie \sigma c}$.

For concept negation $\neg C$,

$$
\begin{aligned}
\mathcal{I} \models a : \neg C \bowtie \sigma c &\Rightarrow (\neg C)^{\mathcal{I}}(a^{\mathcal{I}}) \bowtie \sigma c \\
&\Rightarrow -C^{\mathcal{I}}(a^{\mathcal{I}}) \bowtie \sigma c \\
&\Rightarrow C^{\mathcal{I}}(a^{\mathcal{I}}) \neg \bowtie \sigma \bar{c} \\
&\Rightarrow a^{\mathcal{I}} \in C^{\mathcal{I}'}_{\neg \bowtie \sigma \bar{c}} \\
&\Rightarrow a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\neg \bowtie \sigma \bar{c}} \\
&\Rightarrow \mathcal{I}' \models a : C_{\neg \bowtie \sigma \bar{c}}.
\end{aligned}
$$

For concept conjunction $C \sqcap D$,

$$
\begin{aligned}
\mathcal{I} \models a : C \sqcap D \rhd \sigma c &\Rightarrow (C \sqcap D)^{\mathcal{I}}(a^{\mathcal{I}}) \rhd \sigma c \\
&\Rightarrow \min(C^{\mathcal{I}}(a^{\mathcal{I}}), D^{\mathcal{I}}(a^{\mathcal{I}})) \rhd \sigma c \\
&\Rightarrow (C^{\mathcal{I}}(a^{\mathcal{I}}) \rhd \sigma c) \wedge (D^{\mathcal{I}}(a^{\mathcal{I}}) \rhd \sigma c) \\
&\Rightarrow a^{\mathcal{I}} \in C^{\mathcal{I}'}_{\rhd \sigma c} \wedge a^{\mathcal{I}} \in D^{\mathcal{I}'}_{\rhd \sigma c} \\
&\Rightarrow a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\rhd \sigma c} \wedge a^{\mathcal{I}'} \in D^{\mathcal{I}'}_{\rhd \sigma c} \\
&\Rightarrow a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\rhd \sigma c} \cap D^{\mathcal{I}'}_{\rhd \sigma c} \\
&\Rightarrow \mathcal{I}' \models a : C_{\rhd \sigma c} \sqcap D_{\rhd \sigma c}.
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{I} \models a : C \sqcap D \lhd \sigma c &\Rightarrow (C \sqcap D)^{\mathcal{I}}(a^{\mathcal{I}}) \lhd \sigma c \\
&\Rightarrow \min(C^{\mathcal{I}}(a^{\mathcal{I}}), D^{\mathcal{I}}(a^{\mathcal{I}})) \lhd \sigma c \\
&\Rightarrow (C^{\mathcal{I}}(a^{\mathcal{I}}) \lhd \sigma c) \vee (D^{\mathcal{I}}(a^{\mathcal{I}}) \lhd \sigma c) \\
&\Rightarrow a^{\mathcal{I}} \in C^{\mathcal{I}'}_{\lhd \sigma c} \vee a^{\mathcal{I}} \in D^{\mathcal{I}'}_{\lhd \sigma c} \\
&\Rightarrow a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\lhd \sigma c} \vee a^{\mathcal{I}'} \in D^{\mathcal{I}'}_{\lhd \sigma c} \\
&\Rightarrow a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\lhd \sigma c} \cup D^{\mathcal{I}'}_{\lhd \sigma c} \\
&\Rightarrow \mathcal{I}' \models a : C_{\lhd \sigma c} \sqcup D_{\lhd \sigma c}.
\end{aligned}
$$

For concept disjunction $C \sqcup D$,

$$
\begin{aligned}
\mathcal{I} \models a : C \sqcup D \rhd \sigma c &\Rightarrow (C \sqcup D)^{\mathcal{I}}(a^{\mathcal{I}}) \rhd \sigma c \\
&\Rightarrow \max(C^{\mathcal{I}}(a^{\mathcal{I}}), D^{\mathcal{I}}(a^{\mathcal{I}})) \rhd \sigma c \\
&\Rightarrow (C^{\mathcal{I}}(a^{\mathcal{I}}) \rhd \sigma c) \vee (D^{\mathcal{I}}(a^{\mathcal{I}}) \rhd \sigma c) \\
&\Rightarrow a^{\mathcal{I}} \in C^{\mathcal{I}'}_{\rhd \sigma c} \vee a^{\mathcal{I}} \in D^{\mathcal{I}'}_{\rhd \sigma c} \\
&\Rightarrow a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\rhd \sigma c} \vee a^{\mathcal{I}'} \in D^{\mathcal{I}'}_{\rhd \sigma c} \\
&\Rightarrow a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\rhd \sigma c} \cup D^{\mathcal{I}'}_{\rhd \sigma c} \\
&\Rightarrow \mathcal{I}' \models a : C_{\rhd \sigma c} \sqcup D_{\rhd \sigma c}.
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{I} \models a : C \sqcup D \lhd \sigma c &\Rightarrow (C \sqcup D)^{\mathcal{I}}(a^{\mathcal{I}}) \lhd \sigma c \\
&\Rightarrow \max(C^{\mathcal{I}}(a^{\mathcal{I}}), D^{\mathcal{I}}(a^{\mathcal{I}})) \lhd \sigma c \\
&\Rightarrow (C^{\mathcal{I}}(a^{\mathcal{I}}) \lhd \sigma c) \wedge (D^{\mathcal{I}}(a^{\mathcal{I}}) \lhd \sigma c) \\
&\Rightarrow a^{\mathcal{I}} \in C^{\mathcal{I}'}_{\lhd \sigma c} \wedge a^{\mathcal{I}} \in D^{\mathcal{I}'}_{\lhd \sigma c} \\
&\Rightarrow a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\lhd \sigma c} \wedge a^{\mathcal{I}'} \in D^{\mathcal{I}'}_{\lhd \sigma c} \\
&\Rightarrow a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\lhd \sigma c} \cap D^{\mathcal{I}'}_{\lhd \sigma c} \\
&\Rightarrow \mathcal{I}' \models a : C_{\lhd \sigma c} \sqcap D_{\lhd \sigma c}.
\end{aligned}
$$

For modifier concept δC,

$$\begin{aligned}
\mathcal{I} \models a : \delta C \bowtie \sigma c \;\; &\Rightarrow \;\; \mathcal{I} \models a : C \bowtie \sigma\delta c \\
&\Rightarrow \;\; C^{\mathcal{I}}(a^{\mathcal{I}}) \bowtie \sigma\delta c \\
&\Rightarrow \;\; a^{\mathcal{I}} \in C^{\mathcal{I}'}_{\bowtie\sigma\delta c} \\
&\Rightarrow \;\; a^{\mathcal{I}'} \in C^{\mathcal{I}'}_{\bowtie\sigma\delta c} \\
&\Rightarrow \;\; \mathcal{I}' \models a : C_{\bowtie\sigma\delta c}.
\end{aligned}$$

For universal quantification $\forall R.C$,

$$\begin{aligned}
\mathcal{I} \;\; \models \;\; & a : \forall R.C \rhd \sigma c \\
\Rightarrow \;\; & (\forall R.C)^{\mathcal{I}}(a^{\mathcal{I}}) \rhd \sigma c \\
\Rightarrow \;\; & \inf_{d' \in \Delta^{\mathcal{I}}}\{\max(-R^{\mathcal{I}}(a^{\mathcal{I}}, d'), C^{\mathcal{I}}(d'))\} \rhd \sigma c \\
\Rightarrow \;\; & \textstyle\bigwedge_{d' \in \Delta^{\mathcal{I}}}(\max(-R^{\mathcal{I}}(a^{\mathcal{I}}, d'), C^{\mathcal{I}}(d')) \rhd \sigma c) \\
\Rightarrow \;\; & \textstyle\bigwedge_{d' \in \Delta^{\mathcal{I}}}((-R^{\mathcal{I}}(a^{\mathcal{I}}, d') \rhd \sigma c) \vee (C^{\mathcal{I}}(d') \rhd \sigma c)) \\
\Rightarrow \;\; & \textstyle\bigwedge_{d' \in \Delta^{\mathcal{I}}}((R^{\mathcal{I}}(a^{\mathcal{I}}, d') \neg \rhd \sigma \bar{c}) \vee (C^{\mathcal{I}}(d') \rhd \sigma c)) \\
\Rightarrow \;\; & \forall d' \in \Delta^{\mathcal{I}}.(((a^{\mathcal{I}}, d') \in R^{\mathcal{I}'}_{\neg\rhd\sigma\bar{c}}) \vee (d' \in C^{\mathcal{I}'}_{\rhd\sigma c})) \\
\Rightarrow \;\; & \forall d' \in \Delta^{\mathcal{I}}.(((a^{\mathcal{I}}, d') \notin R^{\mathcal{I}'}_{+\rhd\sigma\bar{c}}) \vee (d' \in C^{\mathcal{I}'}_{\rhd\sigma c})) \\
\Rightarrow \;\; & a^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \forall d' \in \Delta^{\mathcal{I}} : (d, d') \notin R^{\mathcal{I}'}_{+\rhd\sigma\bar{c}} \vee d' \in C^{\mathcal{I}'}_{\rhd\sigma c}\} \\
\Rightarrow \;\; & a^{\mathcal{I}'} = \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'} : (d, d') \notin R^{\mathcal{I}'}_{+\rhd\sigma\bar{c}} \vee d' \in C^{\mathcal{I}'}_{\rhd\sigma c}\} \\
\Rightarrow \;\; & a^{\mathcal{I}'} \in (\forall R_{+\rhd\sigma\bar{c}}.C_{\rhd\sigma c})^{\mathcal{I}'} \\
\Rightarrow \;\; & \mathcal{I}' \models a : \forall R_{+\rhd\sigma\bar{c}}.C_{\rhd\sigma c}.
\end{aligned}$$

$$\begin{aligned}
\mathcal{I} \;\; \models \;\; & a : \forall R.C \lhd \sigma c \\
\Rightarrow \;\; & (\forall R.C)^{\mathcal{I}}(a^{\mathcal{I}}) \lhd \sigma c \\
\Rightarrow \;\; & \inf_{d' \in \Delta^{\mathcal{I}}}\{\max(-R^{\mathcal{I}}(a^{\mathcal{I}}, d'), C^{\mathcal{I}}(d'))\} \lhd \sigma c \\
\Rightarrow \;\; & \textstyle\bigvee_{d' \in \Delta^{\mathcal{I}}}(\max(-R^{\mathcal{I}}(a^{\mathcal{I}}, d'), C^{\mathcal{I}}(d')) \lhd \sigma c) \\
\Rightarrow \;\; & \textstyle\bigvee_{d' \in \Delta^{\mathcal{I}}}((-R^{\mathcal{I}}(a^{\mathcal{I}}, d') \lhd \sigma c) \wedge (C^{\mathcal{I}}(d') \lhd \sigma c)) \\
\Rightarrow \;\; & \textstyle\bigvee_{d' \in \Delta^{\mathcal{I}}}((R^{\mathcal{I}}(a^{\mathcal{I}}, d') \neg \lhd \sigma \bar{c}) \wedge (C^{\mathcal{I}}(d') \lhd \sigma c)) \\
\Rightarrow \;\; & \exists d' \in \Delta^{\mathcal{I}}.(((a^{\mathcal{I}}, d') \in R^{\mathcal{I}'}_{\neg\lhd\sigma\bar{c}}) \wedge (d' \in C^{\mathcal{I}'}_{\lhd\sigma c})) \\
\Rightarrow \;\; & a^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \exists d' \in \Delta^{\mathcal{I}} : (d, d') \in R^{\mathcal{I}'}_{\neg\lhd\sigma\bar{c}} \wedge d' \in C^{\mathcal{I}'}_{\lhd\sigma c}\} \\
\Rightarrow \;\; & a^{\mathcal{I}'} = \{d \in \Delta^{\mathcal{I}'} \mid \exists d' \in \Delta^{\mathcal{I}'} : (d, d') \in R^{\mathcal{I}'}_{\neg\lhd\sigma\bar{c}} \wedge d' \in C^{\mathcal{I}'}_{\lhd\sigma c}\} \\
\Rightarrow \;\; & a^{\mathcal{I}'} \in (\exists R_{\neg\lhd\sigma\bar{c}}.C_{\lhd\sigma c})^{\mathcal{I}'} \\
\Rightarrow \;\; & \mathcal{I}' \models a : \exists R_{\neg\lhd\sigma\bar{c}}.C_{\lhd\sigma c}.
\end{aligned}$$

For existential quantification $\exists R.C$,

$$\begin{aligned}
\mathcal{I} \;\; \models \;\; & a : \exists R.C \rhd \sigma c \\
\Rightarrow \;\; & (\exists R.C)^{\mathcal{I}}(a^{\mathcal{I}}) \rhd \sigma c \\
\Rightarrow \;\; & \sup_{d' \in \Delta^{\mathcal{I}}}\{\min(R^{\mathcal{I}}(a^{\mathcal{I}}, d'), C^{\mathcal{I}}(d'))\} \rhd \sigma c \\
\Rightarrow \;\; & \textstyle\bigvee_{d' \in \Delta^{\mathcal{I}}}(\min(R^{\mathcal{I}}(a^{\mathcal{I}}, d'), C^{\mathcal{I}}(d')) \rhd \sigma c) \\
\Rightarrow \;\; & \textstyle\bigvee_{d' \in \Delta^{\mathcal{I}}}((R^{\mathcal{I}}(a^{\mathcal{I}}, d') \rhd \sigma c) \wedge (C^{\mathcal{I}}(d') \rhd \sigma c)) \\
\Rightarrow \;\; & \exists d' \in \Delta^{\mathcal{I}}.(((a^{\mathcal{I}}, d') \in R^{\mathcal{I}'}_{\rhd\sigma c}) \wedge (d' \in C^{\mathcal{I}'}_{\rhd\sigma c})) \\
\Rightarrow \;\; & a^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \exists d' \in \Delta^{\mathcal{I}} : (d, d') \in R^{\mathcal{I}'}_{\rhd\sigma c} \wedge d' \in C^{\mathcal{I}'}_{\rhd\sigma c}\} \\
\Rightarrow \;\; & a^{\mathcal{I}'} = \{d \in \Delta^{\mathcal{I}'} \mid \exists d' \in \Delta^{\mathcal{I}'} : (d, d') \in R^{\mathcal{I}'}_{\rhd\sigma c} \wedge d' \in C^{\mathcal{I}'}_{\rhd\sigma c}\} \\
\Rightarrow \;\; & a^{\mathcal{I}'} \in (\exists R_{\rhd\sigma c}.C_{\rhd\sigma c})^{\mathcal{I}'} \\
\Rightarrow \;\; & \mathcal{I}' \models a : \exists R_{\rhd\sigma c}.C_{\rhd\sigma c}.
\end{aligned}$$

$$\begin{array}{rcl}
\mathcal{I} & \models & a : \exists R.C \lhd \sigma c \\
& \Rightarrow & (\exists R.C)^{\mathcal{I}}(a^{\mathcal{I}}) \lhd \sigma c \\
& \Rightarrow & \sup_{d' \in \Delta^{\mathcal{I}}}\{\min(R^{\mathcal{I}}(a^{\mathcal{I}}, d'), C^{\mathcal{I}}(d'))\} \lhd \sigma c \\
& \Rightarrow & \bigwedge_{d' \in \Delta^{\mathcal{I}}}(\min(R^{\mathcal{I}}(a^{\mathcal{I}}, d'), C^{\mathcal{I}}(d')) \lhd \sigma c) \\
& \Rightarrow & \bigwedge_{d' \in \Delta^{\mathcal{I}}}((R^{\mathcal{I}}(a^{\mathcal{I}}, d') \lhd \sigma c) \vee (C^{\mathcal{I}}(d') \lhd \sigma c)) \\
& \Rightarrow & \forall d' \in \Delta^{\mathcal{I}}.(((a^{\mathcal{I}}, d') \in R^{\mathcal{I}'}_{\lhd \sigma c}) \vee (d' \in C^{\mathcal{I}'}_{\lhd \sigma c})) \\
& \Rightarrow & \forall d' \in \Delta^{\mathcal{I}}.(((a^{\mathcal{I}}, d') \notin R^{\mathcal{I}'}_{-\lhd \sigma c}) \vee (d' \in C^{\mathcal{I}'}_{\rhd \sigma c})) \\
& \Rightarrow & a^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \forall d' \in \Delta^{\mathcal{I}} : (d, d') \notin R^{\mathcal{I}'}_{-\lhd \sigma c} \vee d' \in C^{\mathcal{I}'}_{\lhd \sigma c}\} \\
& \Rightarrow & a^{\mathcal{I}'} = \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'} : (d, d') \notin R^{\mathcal{I}'}_{-\lhd \sigma c} \vee d' \in C^{\mathcal{I}'}_{\lhd \sigma c}\} \\
& \Rightarrow & a^{\mathcal{I}'} \in (\forall R_{-\lhd \sigma c}.C_{\lhd \sigma c})^{\mathcal{I}'} \\
& \Rightarrow & \mathcal{I}' \models a : \forall R_{-\lhd \sigma c}.C_{\lhd \sigma c}.
\end{array}$$

The proof shows that $\forall \alpha \bowtie \sigma c \in \mathcal{A}.(\mathcal{I} \models \alpha \bowtie \sigma c \Rightarrow \mathcal{I}' \models \theta(\alpha \bowtie \sigma c))$ which implies $\mathcal{I} \models \mathcal{A} \Rightarrow \mathcal{I}' \models \theta(\mathcal{A})$.

(3) Our goal is $\mathcal{I} \models \mathcal{T} \Rightarrow \mathcal{I}' \models \mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) \cup \kappa(\mathfrak{f}\mathcal{K}, \mathcal{T})$.

It is trivial that $\mathcal{I}' \models \mathcal{T}(N^{\mathfrak{f}\mathcal{K}})$ according to our basic idea.

If $C \sqsubseteq D \in \mathcal{T}$, then for all $\sigma c \in \mathcal{N}^{\mathfrak{f}\mathcal{K}}$, $C_{\rhd \sigma c} \sqsubseteq D_{\rhd \sigma c} \in \kappa(\mathfrak{f}\mathcal{K}, \mathcal{T})$ and $D_{\lhd \sigma c} \sqsubseteq C_{\lhd \sigma c} \in \kappa(\mathfrak{f}\mathcal{K}, \mathcal{T})$.

$$\begin{array}{rcl}
\mathcal{I} \models C \sqsubseteq D & \Rightarrow & \forall d \in \Delta^{\mathcal{I}}.C^{\mathcal{I}}(d) \leq D^{\mathcal{I}}(d) \\
& \Rightarrow & \text{if } C^{\mathcal{I}}(d) \rhd \sigma c \text{ then } D^{\mathcal{I}}(d) \rhd \sigma c \\
& \Rightarrow & \text{if } d \in C^{\mathcal{I}'}_{\rhd \sigma c} \text{ then } d \in D^{\mathcal{I}'}_{\rhd \sigma c} \\
& \Rightarrow & C^{\mathcal{I}'}_{\rhd \sigma c} \subseteq D^{\mathcal{I}'}_{\rhd \sigma c} \\
& \Rightarrow & \mathcal{I}' \models C_{\rhd \sigma c} \sqsubseteq D_{\rhd \sigma c}.
\end{array}$$

$$\begin{array}{rcl}
\mathcal{I} \models C \sqsubseteq D & \Rightarrow & \forall d \in \Delta^{\mathcal{I}}.C^{\mathcal{I}}(d) \leq D^{\mathcal{I}}(d) \\
& \Rightarrow & \text{if } D^{\mathcal{I}}(d) \lhd \sigma c \text{ then } C^{\mathcal{I}}(d) \lhd \sigma c \\
& \Rightarrow & \text{if } d \in D^{\mathcal{I}'}_{\lhd \sigma c} \text{ then } d \in C^{\mathcal{I}'}_{\lhd \sigma c} \\
& \Rightarrow & D^{\mathcal{I}'}_{\lhd \sigma c} \subseteq C^{\mathcal{I}'}_{\lhd \sigma c} \\
& \Rightarrow & \mathcal{I}' \models D_{\lhd \sigma c} \sqsubseteq C_{\lhd \sigma c}.
\end{array}$$

So for all $C \sqsubseteq D \in \mathcal{T}$, $\mathcal{I} \models C \sqsubseteq D \Rightarrow \mathcal{I}' \models \{C_{\rhd \sigma c} \sqsubseteq D_{\rhd \sigma c}, D_{\lhd \sigma c} \sqsubseteq C_{\lhd \sigma c}\}$ which implies that $\mathcal{I} \models \mathcal{T} \Rightarrow \mathcal{I}' \models \kappa(\mathfrak{f}\mathcal{K}, \mathcal{T})$. Then we have proved $\mathcal{I} \models \mathcal{T} \Rightarrow \mathcal{I}' \models \mathcal{T}'$.

$\Leftarrow$.) Let $\mathcal{I}'$ be a finite model of $\mathcal{K}(\mathfrak{f}\mathcal{K})$ whose domain $\Delta^{\mathcal{I}'}$ is finite. We build an $\mathcal{ALC}_{\mathcal{FL}}$ interpretation $\mathcal{I}$ such that

- $\Delta^{\mathcal{I}} := \Delta^{\mathcal{I}'}$,
- $a^{\mathcal{I}} := a^{\mathcal{I}'}$ for all individual a,
- $\forall d \in \Delta^{\mathcal{I}}.A^{\mathcal{I}}(d) := \sigma' c'$ for all concept name A, where

  Let $\sigma_1 c_1 = \sup\{\sigma c \mid d \in A^{\mathcal{I}'}_{\rhd \sigma c}\}$, $\sigma_2 c_2 = \inf\{\sigma c \mid d \in A^{\mathcal{I}'}_{\lhd \sigma c}\}$ and $\delta \in H^*$ such that for all $\delta' \in H^*$ and $\delta' \neq \delta$, $\delta' \sigma c > \delta \sigma c > \sigma c$.

  1. Since $\mathcal{K}(\mathfrak{f}\mathcal{K})$ is satisfiable, if $\sigma_1 c_1 = \sigma_2 c_2$ then $\sigma' c' = \sigma_1 c_1 = \sigma_2 c_2$,
  2. otherwise if $\sigma_1 c_1 < \sigma_2 c_2$, $\sigma' c' = \delta \sigma_1 c_1$.

  If $\forall \sigma c.d \notin A^{\mathcal{I}'}_{\bowtie \sigma c}$, $\sigma' c' = \inf(False)$.

- $\forall d, d' \in \Delta^{\mathcal{I}}.R^{\mathcal{I}}(d, d') := \sigma' c'$ for all role name R, where
  Let $\sigma_1 c_1 = \sup\{\sigma c \mid (d, d') \in R^{\mathcal{I}'}_{\triangleright \sigma c}\}$, $\sigma_2 c_2 = \inf\{\sigma c \mid (d, d') \in R^{\mathcal{I}'}_{\triangleleft \sigma c}\}$ and $\delta \in H^*$ such that for all $\delta' \in H^*$ and $\delta' \neq \delta$, $\delta' \sigma c > \delta \sigma c > \sigma c$.

  1. Since $\mathcal{K}(\mathfrak{f}\mathcal{K})$ is satisfiable, if $\sigma_1 c_1 = \sigma_2 c_2$ then $\sigma' c' = \sigma_1 c_1 = \sigma_2 c_2$,
  2. otherwise if $\sigma_1 c_1 < \sigma_2 c_2$, $\sigma' c' = \delta \sigma_1 c_1$.

  If $\forall \sigma c.(d, d') \notin R^{\mathcal{I}'}_{\bowtie \sigma c}$, $\sigma' c' = \inf(False)$.

Because the domain $\Delta^{\mathcal{I}}$ is finite, the interpretation $\mathcal{I}$ is a witnessed interpretation.

(1) We have the following Lemma from our basic idea and the definition of the interpretation $\mathcal{I}$.

Lemma 46. *For all σc and for all $d, d' \in \Delta^{\mathcal{I}'}$, $d \in C^{\mathcal{I}'}_{\bowtie \sigma c} \Rightarrow C^{\mathcal{I}}(d) \bowtie \sigma c$ and $(d, d') \in R^{\mathcal{I}'}_{\bowtie \sigma c} \Rightarrow R^{\mathcal{I}}(d, d') \bowtie \sigma c$.*

Proof. For role name R and concept name A, we obtain $(d, d') \in R^{\mathcal{I}'}_{\bowtie \sigma c} \Rightarrow R^{\mathcal{I}}(d, d') \bowtie \sigma c$ and $d \in A_{\bowtie \sigma c} \Rightarrow A^{\mathcal{I}}(d) \bowtie \sigma c$ from the definition of $\mathcal{I}$ immediately.

Assume for concept term C, $d \in C^{\mathcal{I}'}_{\bowtie \sigma c} \Rightarrow C^{\mathcal{I}}(d) \bowtie \sigma c$ holds.

Let us prove inductively on the structures of concept terms.

For $\neg C$,

$$\begin{aligned} d \in (\neg C)^{\mathcal{I}'}_{\bowtie \sigma c} &\Rightarrow d \in C^{\mathcal{I}'}_{\neg \bowtie \sigma \bar{c}} \\ &\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \neg \bowtie \sigma \bar{c}\} \text{ (by I.H.)} \\ &\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid (\neg C)^{\mathcal{I}}(d') \bowtie \sigma c\} \\ &\Rightarrow (\neg C)^{\mathcal{I}}(d) \bowtie \sigma c. \end{aligned}$$

For δC,

$$\begin{aligned} d \in (\delta C)^{\mathcal{I}'}_{\bowtie \sigma c} &\Rightarrow d \in C^{\mathcal{I}'}_{\bowtie \sigma \delta c} \\ &\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \bowtie \sigma \delta c\} \text{ (by I.H.)} \\ &\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid (\delta C)^{\mathcal{I}}(d') \bowtie \sigma c\} \\ &\Rightarrow (\delta C)^{\mathcal{I}}(d) \bowtie \sigma c. \end{aligned}$$

For $C \sqcap D$,

$$\begin{aligned} d \in (C \sqcap D)^{\mathcal{I}'}_{\triangleright \sigma c} &\Rightarrow d \in (C_{\triangleright \sigma c} \sqcap D_{\triangleright \sigma c})^{\mathcal{I}'} \\ &\Rightarrow d \in C^{\mathcal{I}'}_{\triangleright \sigma c} \cap D^{\mathcal{I}'}_{\triangleright \sigma c} \\ &\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \triangleright \sigma c\} \cap \{d' \in \Delta^{\mathcal{I}'} \mid D^{\mathcal{I}}(d') \triangleright \sigma c\} \text{ (by I.H.)} \\ &\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \triangleright \sigma c \wedge D^{\mathcal{I}}(d') \triangleright \sigma c\} \\ &\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid \min(C^{\mathcal{I}}(d'), D^{\mathcal{I}}(d')) \triangleright \sigma c\} \\ &\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid (C \sqcap D)^{\mathcal{I}}(d') \triangleright \sigma c\} \\ &\Rightarrow (C \sqcap D)^{\mathcal{I}}(d) \triangleright \sigma c. \end{aligned}$$

$$
\begin{aligned}
d \in (C \sqcap D)^{\mathcal{I}'}_{\lhd\sigma c} &\Rightarrow d \in (C_{\lhd\sigma c} \sqcup D_{\lhd\sigma c})^{\mathcal{I}'}\\
&\Rightarrow d \in C^{\mathcal{I}'}_{\lhd\sigma c} \cup D^{\mathcal{I}'}_{\lhd\sigma c}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \lhd \sigma c\} \cup \{d' \in \Delta^{\mathcal{I}'} \mid D^{\mathcal{I}}(d') \lhd \sigma c\} \text{ (by I.H.)}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \lhd \sigma c \vee D^{\mathcal{I}}(d') \lhd \sigma c\}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid \min(C^{\mathcal{I}}(d'), D^{\mathcal{I}}(d')) \lhd \sigma c\}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid (C \sqcap D)^{\mathcal{I}}(d') \lhd \sigma c\}\\
&\Rightarrow (C \sqcap D)^{\mathcal{I}}(d) \lhd \sigma c.
\end{aligned}
$$

For $C \sqcup D$,

$$
\begin{aligned}
d \in (C \sqcup D)^{\mathcal{I}'}_{\rhd\sigma c} &\Rightarrow d \in (C_{\rhd\sigma c} \sqcup D_{\rhd\sigma c})^{\mathcal{I}'}\\
&\Rightarrow d \in C^{\mathcal{I}'}_{\rhd\sigma c} \cup D^{\mathcal{I}'}_{\rhd\sigma c}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \rhd \sigma c\} \cup \{d' \in \Delta^{\mathcal{I}'} \mid D^{\mathcal{I}}(d') \rhd \sigma c\} \text{ (by I.H.)}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \rhd \sigma c \vee D^{\mathcal{I}}(d') \rhd \sigma c\}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid \max(C^{\mathcal{I}}(d'), D^{\mathcal{I}}(d')) \rhd \sigma c\}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid (C \sqcup D)^{\mathcal{I}}(d') \rhd \sigma c\}\\
&\Rightarrow (C \sqcup D)^{\mathcal{I}}(d) \rhd \sigma c.
\end{aligned}
$$

$$
\begin{aligned}
d \in (C \sqcup D)^{\mathcal{I}'}_{\lhd\sigma c} &\Rightarrow d \in (C_{\lhd\sigma c} \sqcap D_{\lhd\sigma c})^{\mathcal{I}'}\\
&\Rightarrow d \in C^{\mathcal{I}'}_{\lhd\sigma c} \cap D^{\mathcal{I}'}_{\lhd\sigma c}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \lhd \sigma c\} \cap \{d' \in \Delta^{\mathcal{I}'} \mid D^{\mathcal{I}}(d') \lhd \sigma c\} \text{ (by I.H.)}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid C^{\mathcal{I}}(d') \lhd \sigma c \wedge D^{\mathcal{I}}(d') \lhd \sigma c\}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid \max(C^{\mathcal{I}}(d'), D^{\mathcal{I}}(d')) \lhd \sigma c\}\\
&\Rightarrow d \in \{d' \in \Delta^{\mathcal{I}'} \mid (C \sqcup D)^{\mathcal{I}}(d') \lhd \sigma c\}\\
&\Rightarrow (C \sqcup D)^{\mathcal{I}}(d) \lhd \sigma c.
\end{aligned}
$$

For $\forall R.C$,

$$
\begin{aligned}
d \in (\forall R.C)^{\mathcal{I}'}_{\rhd\sigma c} &\Rightarrow (\forall.R_{+\rhd\sigma\bar{c}}.C_{\rhd\sigma c})^{\mathcal{I}'}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(d,d') \notin R^{\mathcal{I}'}_{+\rhd\sigma\bar{c}} \vee d' \in C^{\mathcal{I}'}_{\rhd\sigma c}\} \text{ (by I.H.)}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(d,d') \in R^{\mathcal{I}'}_{\neg\rhd\sigma\bar{c}} \vee C^{\mathcal{I}}(d') \rhd \sigma c\}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \textstyle\bigwedge_{d' \in \Delta^{\mathcal{I}'}}(R^{\mathcal{I}}(d,d') \neg \rhd \sigma\bar{c} \vee C^{\mathcal{I}}(d') \rhd \sigma c)\}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \textstyle\bigwedge_{d' \in \Delta^{\mathcal{I}'}}(-R^{\mathcal{I}}(d,d') \rhd \sigma c \vee C^{\mathcal{I}}(d') \rhd \sigma c)\}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \textstyle\bigwedge_{d' \in \Delta^{\mathcal{I}'}}(\max(-R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d')) \rhd \sigma c)\}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \inf_{d' \in \Delta^{\mathcal{I}'}} (\max(-R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d')\} \rhd \sigma c)\}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid (\forall R.C)^{\mathcal{I}}(d) \rhd \sigma c\}\\
&\Rightarrow (\forall R.C)^{\mathcal{I}}(d) \rhd \sigma c.
\end{aligned}
$$

$$
\begin{aligned}
d \in (\forall R.C)^{\mathcal{I}'}_{\lhd\sigma c} &\Rightarrow d \in (\exists R_{\neg\lhd\sigma\bar{c}}.C_{\lhd\sigma c})^{\mathcal{I}'}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \exists d' \in \Delta^{\mathcal{I}'}.(d,d') \in R^{\mathcal{I}'}_{\neg\lhd\sigma\bar{c}} \wedge d' \in C^{\mathcal{I}'}_{\lhd\sigma c}\} \text{ (by I.H.)}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \textstyle\bigvee_{d' \in \Delta^{\mathcal{I}'}}(R^{\mathcal{I}}(d,d') \neg \lhd \sigma\bar{c} \wedge C^{\mathcal{I}}(d') \lhd \sigma c)\}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \textstyle\bigvee_{d' \in \Delta^{\mathcal{I}'}}(-R^{\mathcal{I}}(d,d') \lhd \sigma c \wedge C^{\mathcal{I}}(d') \lhd \sigma c)\}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \textstyle\bigvee_{d' \in \Delta^{\mathcal{I}'}}(\max(-R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d')) \lhd \sigma c)\}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid \inf_{d' \in \Delta^{\mathcal{I}'}} (\max(-R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d')\} \lhd \sigma c)\}\\
&\Rightarrow d \in \{d \in \Delta^{\mathcal{I}'} \mid (\forall R.C)^{\mathcal{I}}(d) \lhd \sigma c\}\\
&\Rightarrow (\forall R.C)^{\mathcal{I}}(d) \lhd \sigma c.
\end{aligned}
$$

For $\exists R.C$,

$$
\begin{array}{rcl}
d \in (\exists R.C)^{\mathcal{I}'}_{\rhd \sigma c} & \Rightarrow & d \in (\exists R_{\rhd \sigma c}.C_{\rhd \sigma c})^{\mathcal{I}'} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid \exists d' \in \Delta^{\mathcal{I}'}.(d,d') \in R^{\mathcal{I}'}_{\rhd \sigma c} \wedge d' \in C^{\mathcal{I}'}_{\rhd \sigma c}\} \text{ (by I.H.)} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid \bigvee_{d' \in \Delta^{\mathcal{I}'}} (R^{\mathcal{I}'}(d,d') \rhd \sigma c \wedge C^{\mathcal{I}}(d') \rhd \sigma c)\} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid \bigvee_{d' \in \Delta^{\mathcal{I}'}} (\min(R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d')) \rhd \sigma c)\} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid \sup_{d' \in \Delta^{\mathcal{I}'}} \{\min(R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d')\} \rhd \sigma c)\} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid (\exists R.C)^{\mathcal{I}}(d') \rhd \sigma c\} \\
& \Rightarrow & (\exists R.C)^{\mathcal{I}}(d) \rhd \sigma c.
\end{array}
$$

$$
\begin{array}{rcl}
d \in (\exists R.C)^{\mathcal{I}'}_{\lhd \sigma c} & \Rightarrow & d \in (\forall R_{-\lhd \sigma c}.C_{\lhd \sigma c})^{\mathcal{I}'} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(d,d') \notin R^{\mathcal{I}'}_{-\lhd \sigma c} \vee d' \in C^{\mathcal{I}'}_{\lhd \sigma c}\} \text{ (by I.H.)} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(d,d') \in R^{\mathcal{I}'}_{\lhd \sigma c} \vee C^{\mathcal{I}}(d') \lhd \sigma c)\} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid \forall d' \in \Delta^{\mathcal{I}'}.(R^{\mathcal{I}}(d,d') \lhd \sigma c \vee C^{\mathcal{I}}(d') \lhd \sigma c)\} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid \bigwedge_{d' \in \Delta^{\mathcal{I}'}} (\min(R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d')) \lhd \sigma c)\} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid \sup_{d' \in \Delta^{\mathcal{I}'}} (\min(R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d')\} \lhd \sigma c)\} \\
& \Rightarrow & d \in \{d \in \Delta^{\mathcal{I}'} \mid (\exists R.C)^{\mathcal{I}}(d') \lhd \sigma c\} \\
& \Rightarrow & (\exists R.C)^{\mathcal{I}}(d) \lhd \sigma c.
\end{array}
$$

□

(2) For ABox, the proof is exactly the reverse processes of that of the ⇒.) from which we can prove $\mathcal{I}' \models \theta(\mathcal{A}') \Rightarrow \mathcal{I} \models \mathcal{A}$ holds.

(3) For all $\sigma c \in \mathcal{N}^{\mathfrak{f}\mathcal{K}}$, $C_{\rhd \sigma c} \sqsubseteq D_{\rhd \sigma c} \in \kappa(\mathfrak{f}\mathcal{K}, \mathcal{T})$, then for all $d \in C^{\mathcal{I}'}_{\rhd \sigma c}$, $d \in D^{\mathcal{I}'}_{\rhd \sigma c}$. Therefore, if $C^{\mathcal{I}}(d) \geq \sigma c$ then $D^{\mathcal{I}}(d) \geq \sigma c$.

Assume $\mathcal{I}' \models \mathcal{T}'$ and $\mathcal{I} \not\models C \sqsubseteq D$ where $C \sqsubseteq D \in \mathcal{T}$. So there exists a $d' \in \Delta^{\mathcal{I}}$ such that $C^{\mathcal{I}}(d') > D^{\mathcal{I}}(d')$. Consider $C^{\mathcal{I}}(d') = \sigma' c'$. Of course $C^{\mathcal{I}}(d') \geq \sigma' c'$. Therefore, $D^{\mathcal{I}}(d') \geq \sigma' c'$. From the hypothesis it follows that $\sigma' c' = C^{\mathcal{I}}(d') > D^{\mathcal{I}}(d') \geq \sigma' c'$, which contradicts the hypothesis. So $\mathcal{I}' \models \mathcal{T}' \Rightarrow \mathcal{I} \models \mathcal{T}$. □

Chapter 4

Implementation

In this section, we demonstrate the implementation of the transformation. The input of the program is an OWL-DL ontology with an $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base and the output is a classical OWL-DL ontology with an $\mathcal{ALCH}$ knowledge base. The OWL-DL Ontology and the KAON2 will be briefly explained soon.

4.1 OWL ontology

4.1.1 Introduction to OWL ontology

The term *ontology* is borrowed from philosophy in which an ontology is a theory about the nature of existence, of what types of things exist. The Semantic Web is a vision for the future of the Web in which information is given explicit meaning, making it easier for machines to automatically process and integrate information available on the Web. While as a basic component of the Semantic Web, an ontology is a collection of information and is a document or file that formally defines the relations among terms.

OWL[1] is a *Web Ontology Language* and is intended to provide a language that can be used to describe the classes and relations between them that are inherent in Web documents and applications. OWL is a vocabulary extension of RDF.[2] The OWL language provides three increasingly expressive sublanguages: OWL Lite, OWL DL, OWL Full. The input of our transformation program is an OWL DL ontology. OWL DL is so named due to its correspondence with description logics. OWL DL was designed to support the existing Description Logic business segment and has desirable computational properties for reasoning systems. According to the corresponding relation between axioms of OWL ontology and terms of Description Logic, we can represent the knowledge base contained in the ontology in syntax of DLs.

In order to make the Semantic Web be able to handle applications that face uncertain and imprecise information, many researchers have proposed extending OWL and

[1]Please visit http://www.w3.org/TR/owl-guide/ for more details.
[2]http://www.w3.org/RDF/

Description Logic to deal with such uncertainty. f-OWL,[3] a fuzzy extension to OWL, can capture imprecise and vague knowledge. The accompanying Fuzzy Reasoning Engine lets f-OWL capture and reason about such knowledge. So far, OWL does not have any syntax for hedges. In the following section, I develop an rdfs-based syntax for hedges.

4.1.2 A syntax for hedge

The following is an rdfs-based syntax for hedges. This Axiom represents a modifier concept $VeryC$ where $Very$ is a hedge and C is a concept term.

```
<owl:Class rdf:about="VeryC">
 <owl:Hedge> Very </owl:Hedge>
 <owl:modifierOf>
    <owl:Class rdf:about="http://www.w3.org/2002/07/owl#C"/>
 </owl:modifierOf>
</owl:Class>
```

Table 1: a rdfs-based syntax for hedge

Example 47. *For a concept $VeryMoreA$,*
<owl:Class rdf:about="VeryMoreA">
<owl:Hedge> Very </owl:Hedge>
<owl:modifierOf>
<owl:Class rdf:about="http://www.w3.org/2002/07/owl#MoreA"/>
<owl:Hedge> More </owl:Hedge>
<owl:modifierOf>
<owl:Class rdf:about="http://www.w3.org/2002/07/owl#A"/>
</owl:modifierOf>
</owl:Class>
</owl:modifierOf>
</owl:Class>

4.2 Introduction to KAON2

KAON2[4] supports ontology languages. It is based on OWL and can read OWL syntax, so we use KAON2 to manipulate OWL-DL ontologies. Table 2 gives the classes of KAON2 we mainly use and their corresponding syntax in $\mathcal{ALCH}$.

From Table 2, we know that `ClassMember` is concept assertion and `ObjectProperty-Member` is role assertion. `SubClassOf, EquivalentClasses` are concept terminological axioms. `Description` corresponds to concept term.

[3] http://fowl.sourceforge.net/about.html

[4] KAON2 is an infrastructure for managing OWL ontologies. The API of KAON2 is capable of manipulating OWL ontologies. Please visit http://kaon2.semanticweb.org/ for more details about how to use KAON2.

KAON2 Class	DLs Syntax	Example
`Individual`	individual name	a, b
`thing`	top concept	$\top$
`nothing`	bottom concept	$\bot$
`OWLClass`	concept name	A
`ObjectProperty`	role name	R
`Description`	concept term	C, D
`ObjectAnd`	concept conjunction	$C \sqcap D$
`ObjectOr`	concept disjunction	$C \sqcup D$
`ObjectNot`	concept negation	$\neg C$
`ObjectSome`	existential quantification	$\exists R.C$
`ObjectAll`	universal quantification	$\forall R.C$
`ClassMember`	concept assertion	$a : C$
`ObjectPropertyMember`	role assertion	$(a, b) : R$
`SubClassOf`	concept inclusion	$C \sqsubseteq D$
`SubObjectPropertyOf`	role inclusion	$R \sqsubseteq S$

Table 2: OWL Descriptions

KAON2 implements a resolution-based decision procedure for general TBoxes (subsumption, satisfiability, classification) and ABoxes (retrieval, conjunctive query answering). KAON2 treats all of OWL DL except nominals. Generally speaking, KAON2 seems to do better on Abox reasoning tasks than reasoners which implement the tableaux calculus do, in particular if ABox is large and TBox is of medium size [8].

4.3 Introduction to Jena

Jena[5] is a Java framework for building Semantic Web applications. It provides a programmatic environment for RDF, RDFS and OWL, SPARQL[6] and includes a rule-based inference engine. It is a pretty writer for RDF. Jena can load and write ontologies with syntax that KAON2 can not handle, e.g., the rdf-based syntax of hedges. For this reason, I use Jena to load and read the ontologies with hedges.

4.4 Introduction to Racer

RACER[7] stands for Renamed ABox and Concept Expression Reasoner. RacerPro is the commercial name of the software. It is a lisp-based reasoner. It appeared in 2002 and have been continuously improved. RacerPro is a knowledge representation system that implements a highly optimized tableau calculus for a very expressive description logic.

[5]http://jena.sourceforge.net/
[6]http://www.w3.org/TR/rdf-sparql-query/
[7]http://www.racer-systems.com/

With the exception of nominals, which are very hard to optimize, RacerPro supports the full OWL standard. Racer provides reasoning support for instances, consistency checking and the subsumption problem.

To access RacerPro from our own application we can use these APIs to the native command set of RacerPro. The libraries utilize the TCP/IP interface to access RacerPro. Users of the Desktop or a time-limited trial or educational license of RacerPro can only use these APIs on their local machine (local computation only).

4.5 The algorithm

The implementation goes as the following algorithm.

input: an $\mathcal{ALC}_{\mathcal{FL}}$ knowledge base $\mathfrak{f}\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$
output: an $\mathcal{ALCH}$ knowledge base $\mathcal{K} = \langle \mathcal{T}', \mathcal{A}' \rangle$
variables: $i, j, k \in \mathbb{N}$
begin
$\mathcal{A}' := \emptyset$;
$\mathcal{T}' := \emptyset$;
$\mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) := \emptyset$;
$\kappa := \emptyset$;
$N^{\mathfrak{f}\mathcal{K}} := \{n_1, \ldots, n_{|N^{\mathfrak{f}\mathcal{K}}|}\}$;
$A^{\mathfrak{f}\mathcal{K}} := \{A \mid A$ is a concept name and occurs in $\mathfrak{f}\mathcal{K}\}$;
$R^{\mathfrak{f}\mathcal{K}} := \{R \mid R$ is a role name and occurs in $\mathfrak{f}\mathcal{K}\}$;
for every $\langle \alpha \bowtie \sigma c \rangle \in \mathcal{A}$ do $\mathcal{A}' := \mathcal{A}' \cup \{\theta(\alpha \bowtie \sigma c)\}$;
for every $C \sqsubseteq D \in \mathcal{T}$ do
    for k := 1 to $|N^{\mathfrak{f}K}|$ do
        $\kappa := \kappa \cup \{\rho(C, > n_k) \sqsubseteq \rho(D, > n_k), \rho(C, \geq n_k) \sqsubseteq \rho(D, \geq n_k),$
        $\rho(D, < n_k) \sqsubseteq \rho(C, < n_k), \rho(D, \leq n_k) \sqsubseteq \rho(C, \leq n_k)\}$;
for every $A \in \mathcal{A}^{\mathfrak{f}\mathcal{K}}$ do
    begin
        for i := 1 to $|N^{\mathfrak{f}K}| - 1$ do
            $\mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) := \mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) \cup \{A_{\geq n_{i+1}} \sqsubseteq A_{>n_i}, A_{>n_i} \sqsubseteq A_{\geq n_i}, A_{\leq n_i} \sqsubseteq A_{<n_{i+1}},$
            $A_{>n_i} \sqcap A_{\leq n_i} \sqsubseteq \bot, \top \sqsubseteq A_{>n_i} \sqcup A_{\leq n_i}\}$;
        for j := 2 to $|N^{\mathfrak{f}K}|$ do
            $\mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) := \mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) \cup \{A_{<n_j} \sqsubseteq A_{\leq n_j}, A_{\geq n_j} \sqcap A_{<n_j} \sqsubseteq \bot,$
            $\top \sqsubseteq A_{\geq n_j} \sqcup A_{<n_j}\}$;
        for i := 1 to $|N^{\mathfrak{f}K}| - 1$ do
            $\mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) := \mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) \cup \{R_{\geq n_{i+1}} \sqsubseteq R_{>n_i}, R_{>n_i} \sqsubseteq_{\geq n_i}\}$;
    end
$\mathcal{T}' := \mathcal{T}(N^{\mathfrak{f}\mathcal{K}}) \cup \kappa$;
$\mathcal{K} := \langle \mathcal{T}', \mathcal{A}' \rangle$;
end.

Table 3. algorithm of transformation

The algorithm encodes the transformation presented previously and is implemented with the programming language JAVA, the API of Jena and KAON2. The complexity of this algorithm is polynomial time of the size of the knowledge base.

4.6 Testing

The data used to do the test is a number of ontologies which are generated by a program written with programming JAVA and API of Jena. Since no existing reasoner generates fuzzy knowledge bases and no fuzzy ontology is available yet, a program is written to this end. Although KAON2 manipulates OWL, it can not create and load the ontology with the syntax for hedge which is shown in Table 1. The API of Racer is not good at manipulating ontology, while Jena can do all of these easily, so a JAVA program using Jena to create ontologies is implemented. Please refer to Appendix B for the detail of how to produce fuzzy ontologies with the program.

To test this algorithm, we generate a number of ontologies whose satisfiabilities are known. After transforming them, we use KAON2 and Racer to test whether the output ontologies have the same satisfiabilities with the input ontologies. The transformation program shows the satisfiabilities of knowledge bases right after transforming them by using KAON2.

4.7 KAON2 vs Racer

KAON2 and Racer are both excellent reasoners. Since they are based on different algorithm, they have their own advantages and have different optimizations. KAON2 is resolution-based and Racer is tableau-based. KAON2 provides a stand-alone server providing access to ontologies, while Racer supplies a client which offers a lean interface to communicate with RacerPro via HTTP in the DIG protocol.

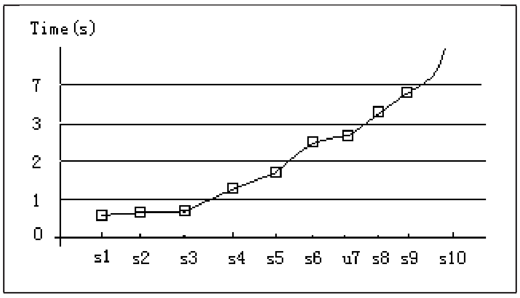

Figure 4.1: the performance of KAON2

It took KAON2 0.611 second on average to answer the satisfiability of a small ontology like s1.owl which has a small TBox and an ABox with only one assertion. KAON2 spent more and more time along with the increasing of sizes of TBoxes. KAON2 used 6.99 seconds on average to do the same reasoning for ontologies (s9.owl) who have normal size ABoxes and big TBoxes. We can see that KAON2 does better on Abox reasoning tasks, but not good at dealing with TBox. In our case, the size of TBox is usually not small. We can see in the Figure 4.1 , the bigger the size of a knowledge base is, the more time it took KAON2 to do the reasoning.

KAON2 does not perform a separate ABox consistency test because ABox inconsistency is discovered automatically during query evaluation. Figure 4.2 shows the Tbox-consistency times for these ontologies. The results are sorted based on the size of the TBox. Figure 4.2 shows that the performance of TBox reasoning in Racer lags behind the performance of KAON2 when TBox is of medium size.

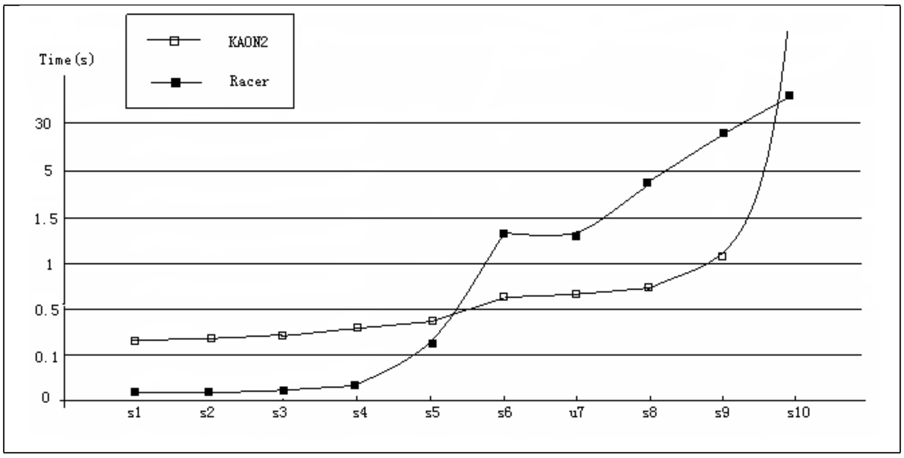

Figure 4.2: KAON2 vs Raver

Furthermore, when one uses JAVA, KAON2 is a much better tool to manipulate ontologies than Racer.

Chapter 5

Conclusions

We have presented a satisfiability preserving transformation of $\mathcal{ALC}_{\mathcal{FL}}$ into $\mathcal{ALCH}$. Soundness and completeness of the transformation have been shown.

The difference between this transformation and the related work [14] which transform fuzzy $\mathcal{ALCH}$ into classical $\mathcal{ALCH}$ is that truth domains of $\mathcal{ALC}_{\mathcal{FL}}$ and fuzzy $\mathcal{ALCH}$ are different. $\mathcal{ALC}_{\mathcal{FL}}$ uses hedges as the fuzzy extension and the truth domain of interpretations is represented by a hedge algebra. Moreover, the hedges occur not only in the fuzzy values but also in concept terms. Thus there is one more rule for dealing with modifier concept terms.

We have the same restriction as in [14] that fuzzy assertions of the form $(a.b) : R < \sigma c$ or $(a.b) : R \leq \sigma c$ are not allowed because they related to 'negated role' which is not part of classical $\mathcal{ALCH}$. In order to insure the satisfiability preserving property, we consider only witnessed interpretations of $\mathcal{ALC}_{\mathcal{FL}}$.

As far as I know, there are a few reasoners for fuzzy description logics, e.g., fuzzyDL[1] and FiRE, [2] but no reasoner for fuzzy description logics with hedges. The reasoning in $\mathcal{ALC}_{\mathcal{FL}}$ can employ already existing DL systems by transforming it into $\mathcal{ALCH}$.

As for the complexity of the transformation, we know that,

1. $|\theta(\mathcal{A})|$ is linearly bounded by $|\mathcal{A}|$;
2. $|\mathcal{T}(N^{\mathfrak{f}\mathcal{K}})| = 8|\mathcal{A}^{\mathfrak{f}\mathcal{K}}|(|\mathcal{N}^{\mathfrak{f}\mathcal{K}}| - 1) + 2|\mathcal{R}^{\mathfrak{f}\mathcal{K}}|(|\mathcal{N}^{\mathfrak{f}\mathcal{K}}| - 1)$;
3. $\kappa(\mathfrak{f}\mathcal{K}, \mathcal{T})$ contains at most $4|\mathcal{T}||N^{\mathfrak{f}\mathcal{K}}|$.

Therefore, the resulted classical knowledge base (at most polynomial size) can be constructed in polynomial time.

If $\mathcal{T}$ is an acyclic TBox, we can reduce reasoning problems with respect to $\mathcal{T}$ to problems with respect to the empty TBox. That is good for KAON2, as the developing staff claim KAON2 appears to be inferior on TBox reasoning tasks [8]. Unfortunately, expanding TBox increases the complexity of reasoning up to Exp-Time. If one allows cyclic terminological axioms the expanding of TBox may not terminate. It is even

[1] http://gaia.isti.cnr.it/~straccia/

[2] http://www.image.ece.ntua.gr/~nsimou/

worse in our case, because $\mathcal{ALC}_{\mathcal{FL}}$ allows concept inclusions for which the expanding is not applicable [1]. For concept inclusions, the expanding is not applicable, because the expanding requires the left of the concept definition [1] is primitive concept, but concept inclusions can not insure this condition, and a concept inclusion might be a cyclic terminological axiom.

The extension of this work can be transformations of fuzzy DLs with different truth domains of interpretations into classical DLs.

Appendix A

An example of running the program

In this section, we use an example to test the program. Consider the knowledge base of Example 31,

$$\begin{aligned}\mathcal{K} \quad = \quad & \{\langle Tom : \neg(\exists weight.VeryVeryHeavy) \sqcup Fat \geq True\rangle, \\ & \langle (Tom, 150) : weight \geq MolTrue\rangle, \\ & \langle 150 : Heavy \geq MoreTrue\rangle\}.\end{aligned}$$

We want to know whether Tom is fat, i.e., whether the following holds,

$$\mathcal{K} \models \langle Tom : Fat \geq True\rangle. \tag{A.1}$$

We can reduce this problem to the satisfiability of the knowledge base $\mathcal{K}'$, such that $\mathcal{K}' = \mathcal{K} \cup \{\langle Tom : Fat < True\rangle\}$. If $\mathcal{K}'$ is unsatisfiable then A.1 holds.

Let's use $\mathcal{K}'$ as the input knowledge base, the program transformed the knowledge base and listed the $\mathcal{ALCH}$ knowledge base as the result (see Figure A.1). After that, a message is shown by a reasoner called by the program which told us "The knowledge base is unsatisfiable". Thus A.1 holds. We conclude that Tom is fat.

The following is the new ABox,

$$\begin{aligned}& \{\langle Tom : (\forall weight_{>False}.Heavy_{\leq VeryVeryFalse}) \sqcup Fat_{\geq True}\rangle, \\ & \langle (Tom, 150) : weight_{\geq MolTrue}\rangle, \\ & \langle 150 : Heavy_{\geq MoreTrue}\rangle, \\ & \langle Tom : Fat_{<} True\rangle\}.\end{aligned}$$

TBox is quite large so we just give a small part of it,

$$\begin{aligned}& \{Fat_{\geq True} \sqsubseteq Fat_{>MoLTrue}, \\ & Fat_{>MoLTrue} \sqsubseteq Fat_{\geq MoLTrue}, \\ & Fat_{\leq MoLTrue} \sqsubseteq Fat_{<True}, \\ & Fat_{>MoLTrue} \sqcap Fat_{\leq MoLTrue} \sqsubseteq \bot, \\ & \top \sqsubseteq Fat_{>MoLTrue} \sqcup Fat_{\leq MoLTrue}, \ldots\}.\end{aligned}$$

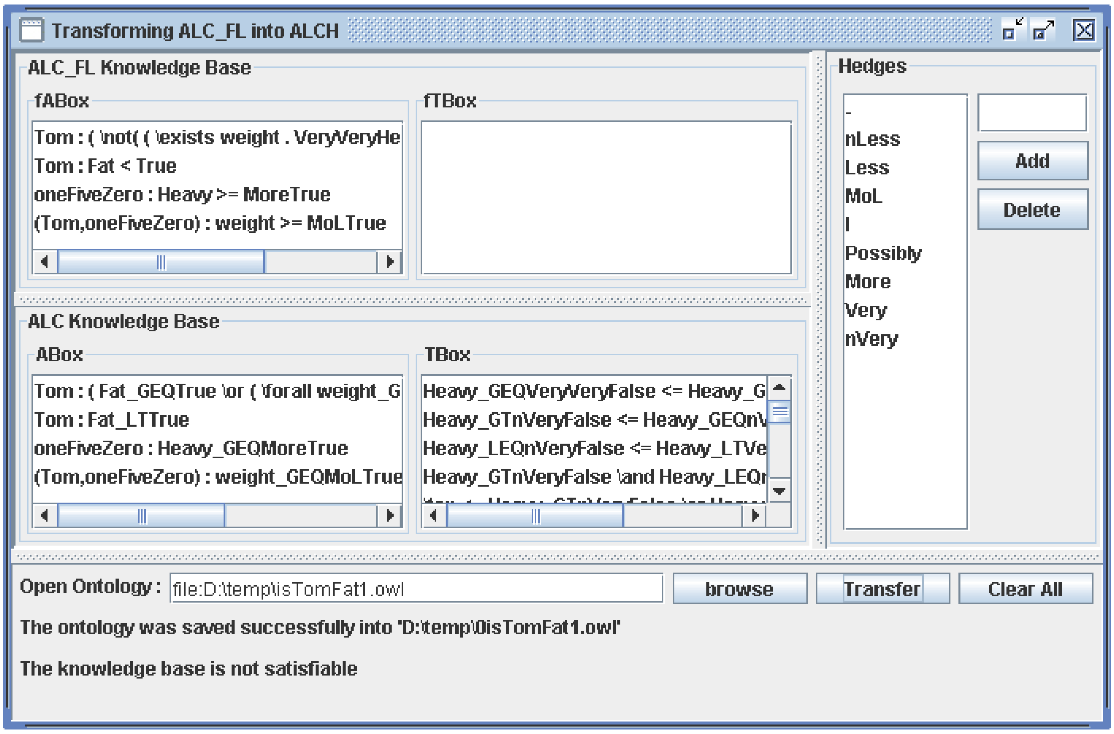

Figure A.1: The transforming application

Appendix B

The implementation of producing $\mathcal{ALC}_{\mathcal{FL}}$ knowledge bases

We introduce the program which produces fuzzy ontologies with $\mathcal{ALC}_{\mathcal{FL}}$ knowledge bases (Figure B.3). We can produce concept assertions, role assertions and inclusions by using this program. Individuals, concept names, role names and hedges can be added.

To build a concept term, the constructor should be chosen first from the first Combo box, operands are selected next.

Example 48. *“\not \and Very C \forall R D” represents* $\neg((VeryC) \sqcap (\forall R.D))$.

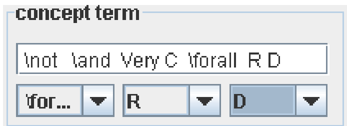

Figure B.1: The concept term panel

Given a HA $AX = (X, G, H, >)$, if the greatest hedge in H is $Very$, we represent $\sup(True)$ as $nVery(True)$ and $\inf(False)$ as $nVery(False)$ respectively where we take $nVery$ as the greatest element of H^*. If the least hedge in H is $Less$, we express $\inf(True)$ as $nLess(True)$ where $nLess$ is the least element of H^*.

We input chains of hedges of the form “$H_n\_\ldots\_H_3\_H_2\_H_1\_$”. There must be a “_” between any two hedges. The last character should be a “_”. In Figure B.2, “$Very\_More\_$” is a chain of hedges of the correct form.

Figure B.2: the TextField of hedge chains

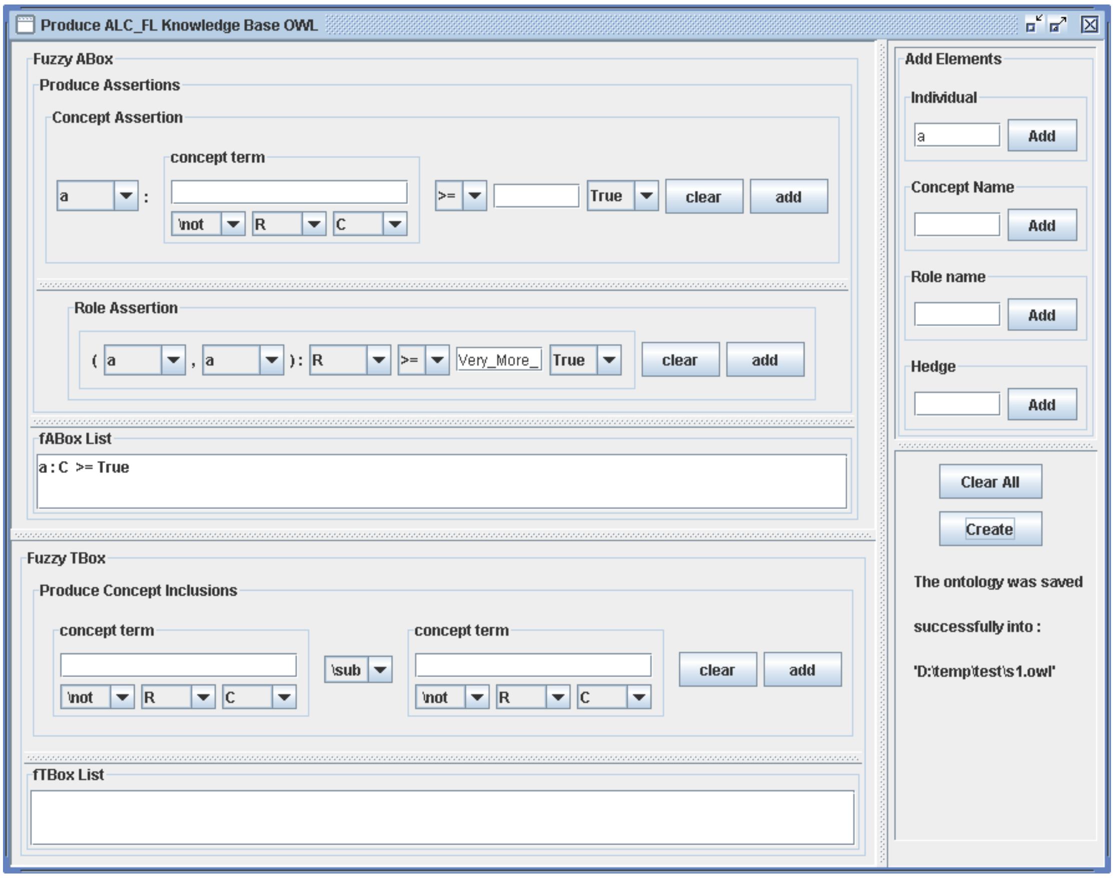

Figure B.3: The producer

Bibliography

[1] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[2] Petr Hájek. Making fuzzy description logic more general. *Fuzzy Sets and Systems*, 154(1):1–15, 2005.

[3] Petr Hájek. What does mathematical fuzzy logic offer to description logic. In *Fuzzy Logic and the Semantic Web*, 2006.

[4] N. Cat Ho and W. Wechler. Hedge algebras: an algebraic approach to structure of sets of linguistic truth values. *Fuzzy Sets and Systems*, 35(3):281–293, 1990.

[5] Steffen Hölldobler, Dinh-Khac Dzung, and Tran Dinh-Khang. The fuzzy linguistic description logic $\mathcal{ALC}_{\mathcal{FL}}$. In *Proceedings of the Eleventh International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, pages 2096–2103, 2006.

[6] Steffen Hölldobler, Tran Dinh Khang, and Hans-Peter Störr. A fuzzy description logic with hedges as concept modifiers. In Nguyen Hoang Phuong, Hung T. Nguyen, Nguyen Cat Ho, and Pratit Santiprabhob, editors, *Proceedings InTech/VJFuzzy'2002*, pages 25–34, Hanoi, Vietnam, 2002. Institute of Information Technology, Vietnam Center for Natural Science and Technology, Science and Technics Publishing House, Hanoi, Vietnam.

[7] Steffen Hölldobler, Hans-Peter Störr, and Dinh Khang Tran. The fuzzy description logic $\mathcal{ALC}_{\mathcal{FH}}$ with hedge algebras as concept modifiers. *Journal of Advanced Computational Intelligence and Intelligent Informatics (JACIII)*, 7(3):294–305, 2003.

[8] Boris Motik and Ulrike Sattler. A comparison of reasoning techniques for querying large description logic aboxes. In *LPAR*, pages 227–241, 2006.

[9] Cat-Ho Nguyen, Dinh-Khang Tran, Van-Nam Huynh, and Hai-Chau Nguyen. Hedge algebras, linguistic-valued logic and their application to fuzzy reasoning. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 7(4):347–361, 1999.

[10] Cat-Ho Nguyen and W. Wechler. Extended hegde algebras and their application to fuzzy logic. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 52:259–281, 1992.

[11] H.-N. Nguyen S. Hölldobler and D.-K. Tran. The fuzzy description logic $\mathcal{ALC}_{\mathcal{FLH}}$. In *Proc. 9th IASTED International Conference on Artificial Intelligence and Soft Computing*, pages 99–104, 2005.

[12] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with complements. *Artif. Intell.*, 48(1):1–26, 1991.

[13] Umberto Straccia. Reasoning within fuzzy description logics. *J. Artif. Intell. Res. (JAIR)*, 14:137–166, 2001.

[14] Umberto Straccia. Transforming fuzzy description logics into classical description logics. In *Proceedings of the 9th European Conference on Logics in Artificial Intelligence (JELIA-04)*, number 3229 in Lecture Notes in Computer Science, pages 385–399, Lisbon, Portugal, 2004. Springer Verlag.

[15] Lotfi A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning - i. *Information Sciences*, 8(3):199–249, 1975.

[16] Lotfi A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning - ii. *Information Sciences*, 8(4):301–357, 1975.

[17] Lotfi A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning-iii. *Information Sciences*, 9(1):43–80, 1975.