RWTH
LTCS–Report

Aachen University of Technology
Research group for
Theoretical Computer Science

A Description Logic for Vague Knowledge

Christopher B. Tresp Ralf Molitor

LTCS-Report 98-01

An abridged version of this paper has been published in the Proceedings of the 13th biennial European Conference on Artificial Intelligence (ECAI'98).

RWTH Aachen
LuFg Theoretische Informatik
http://www-lti.informatik.rwth-aachen.de

Ahornstr. 55
52074 Aachen
Germany

A Description Logic for Vague Knowledge

C.B. Tresp and R. Molitor
RWTH-Aachen, LuFg Theoretical Computer Science
Ahornstr. 55, 52074 Aachen, Germany

Abstract

This work introduces the concept language $\mathcal{ALC}_{F_M}$ which is an extension of $\mathcal{ALC}$ to many-valued logics. $\mathcal{ALC}_{F_M}$ allows to express vague concepts, e.g. *more or less enlarged* or *very small*. To realize this extension to many-valued logics, the classical notions of satisfiability and subsumption had to be modified appropriately. For example, $\mathcal{ALC}_{F_M}$-concepts are no longer either satisfiable or unsatisfiable, but they are satisfiable to a certain degree. The main contribution of this paper is a sound and complete method for computing the degree of subsumption between two $\mathcal{ALC}_{F_M}$-concepts.

1 Introduction

This work takes its motivation from the occurrence of vague concept descriptions in different application areas. Often, application–inherent information is characterized by a very high degree of vagueness. Appropriate information systems must be able to process this kind of data. So far, there are no systems that really solve the corresponding problems due to the lack of powerful basic methods.

A comfortable base of an information system can be obtained by a terminological knowledge representation system (TKRS), originated by Brachman and Schmolze in [2]. The underlying knowledge representation formalism provides the user with a concept language to formulate and to solve application–relevant problems. Especially the advantage of symbolic representation allows the transparent acquisition and processing of information. But TKRS mostly lack the possibility to represent vague concept definitions in an appropriate manner.

Current approaches like [8, 7, 6] most often incorporate uncertainty calculus into the formalism of the TKRS. Uncertainty calculi deal with assumptions based on previous observations. These assumptions lack the feature of definitive evidence and thus they must not be complete. The appearance of vagueness addresses a different problem (see [3]). Though the information is complete vagueness arises from the existence of knowledge areas with concepts that have no clear demarcation among others. The borderline between the definitions of these concepts is rather hazy. For example, a medical statement that connects

a patient's temperature with some linguistic category like *high* or *medium* is based on concrete (and therefore complete) measured values. Questions are to determine the *membership degree* of values (like 37.9^oC) to categories/sets and especially to infer knowledge from vague concepts.

Therefore, it seems profitable, to provide TKRS with methods from the discipline of many-valued logics to open a gateway for vague information processing.

This article is structured as follows. Normally, concept languages are based on fragments of first order predicate calculus. The underlying first order logic is extended to a many-valued logic first. Using these extensions, syntax and semantics of the concept language are defined. Then, an appropriate subsumption algorithm for the new language is introduced and its main properties are proved. The work concludes with some notes on further extensions.

2 Logical Basis

A signature Σ is defined as usual with a sort symbol s, a set FS of constant symbols and a set PS of predicate symbols.

Let JM be a fixed set of connectors $\{\wedge_{\mathsf{min}}, \vee_{\mathsf{max}}, M_1, \ldots, M_n\}$, where all M_i are unary connectors. The negation $\neg$ corresponds to some unary connector M_i. Let further QM be a fixed set of quantifiers $\{\exists_{Sup}, \forall_{Inf},\}$. An expression is defined as follows:

Let V denote an infinite set of (individual) *variables*. *Terms* (constant and variable symbols) and *expressions* are introduced as usual, where all connectors from JM are allowed to form expressions. First order predicate logic quantifiers $\forall$ and $\exists$ are replaced by $\forall_{Inf}$ and $\exists_{Sup}$. A *weighted expression* is an expression C together with a truth value from the unit interval, written $[C, t]$.

Definition 1 (Interpretation) *A tuple* $\mathfrak{A} = [\Sigma, \Delta, \Phi, \mu]$ *is called* interpretation *of* Σ*, if and only if it satisfies the following conditions:*

1. *The sort symbol s is interpreted as a non-empty set of individuals Δ. Single individuals are denoted as ξ.*
2. *Φ is a mapping that maps each constant c in FS to a $\Phi_c \in \Delta$.*
3. *μ is a mapping that maps each n-ary predicate symbol $P \in PS$ to a total function $\mu_P : \Delta^n \rightarrow [0..1]$.*

Definition 2 (Allocation of variables) 1. *An $\mathfrak{A}$-allocation σ is a mapping $\sigma : V \rightarrow \Delta$ of variables in V to elements in Δ.*

2. *$EL(T, \mathfrak{A}, \sigma)$ is the element of Δ, connected to the term T by $\mathfrak{A}$ under $\mathfrak{A}$-allocation σ. Therefore, it is $EL(x, \mathfrak{A}, \sigma) = \sigma(x)$, if $x \in V$ and $EL(c, \mathfrak{A}, \sigma) = \Phi_c$, if $c \in FS$.*

3. *For a given $\mathfrak{A}$-allocation σ and a fixed variable $x_i \in V$ and a fixed individual ξ_i in Δ, $\sigma\langle x_i := \xi_i\rangle$ denotes the $\mathfrak{A}$-allocation σ', defined for an*

arbitrary $x \in V$ *as*

$$\sigma'(x) = \begin{cases} \sigma(x) & \text{for } x \neq x_i \\ \xi_i & \text{for } x = x_i \end{cases}$$

The realization of a connector J_i from JM is a mapping $j_i : [0..1]^n \to [0..1]$. As an example, $\wedge_{\mathsf{min}}$ is realized by the function min (minimum of two values). The realization of a quantifier Q_i is a mapping $q_i : \mathcal{P}([0..1]) \to [0..1]$ (where $\mathcal{P}([0..1])$ is the set of all subsets of $[0..1]$). For example, $\forall_{inf}$ is realized by the *Infimum* of a set of truth values. As stated above, $\neg$ is one of the connectors M_i. They are called *manipulators*. Their possible realizations (e.g. the function $m_i(t) = 1 - t$ for $M_i = \neg$, $t \in [0..1]$) are formally defined within the next section.

Definition 3 (Satisfiability of Weighted Expressions)

1. $\sigma, \mathfrak{A} \models_{t'} PT_1 \ldots T_n$ *iff*
   $t' = \mu_P(EL(T_1, \mathfrak{A}, \sigma), \ldots, EL(T_n, \mathfrak{A}, \sigma))$,
2. $\sigma, \mathfrak{A} \models_{t'} J_i(C_1, \ldots, C_n)$ *iff* *exist* $t_1, \ldots, t_n \in [0..1]$ *such that* $\sigma, \mathfrak{A} \models_{t_j} C_j$, $1 \leq j \leq n$, *and* $t' = j_i(t_1, \ldots, t_n)$,
3. $\sigma, \mathfrak{A} \models_{t'} Q_i\, x_i : D$ *iff*
   $t' = q_i\{t \mid$ *Exists* $\xi_i \in \Delta$ *such that* $\sigma\langle x_i := \xi_i\rangle, \mathfrak{A} \models_t D\}$.

In the sequel, the notion $\sigma, \mathfrak{A} \models [C, t]$ will be used instead of $\sigma, \mathfrak{A} \models_t C$.

For values $t_{C_1}, \ldots, t_{C_n} \in [0..1]$ the set $\{[C_1, t_{C_1}], \ldots, [C_n, t_{C_n}]\}$ is *satisfiable* if and only if there exists an $\mathfrak{A}$-allocation σ such that $\sigma, \mathfrak{A} \models [C_i, t_{C_i}]$ for all $1 \leq i \leq n$.

For a fixed signature Σ, the notion of semantic consequence is important:

Definition 4 (Consequence)
C *entails* D *to degree* t_D ($C \models [D, t_D]$) *iff*

$$t_D = Inf\{t' \mid \mathfrak{A} \text{ interpretation of } \Sigma, \sigma\, \mathfrak{A}\text{-allocation with } \sigma, \mathfrak{A} \models_1 C \text{ and } \sigma, \mathfrak{A} \models_{t'} D\}.$$

As a precondition the truth degree of the premise C has to be 1 since it seems reasonable to infer vague knowledge from valid premises (fulfilled to the degree 1). If C is not satisfiable to degree 1, D is entailed to degree 1 analogous to the consequence relation in classic first order logic. Therefore, let $Inf(\emptyset) := 1$.

3 The Description Language

Within the language, it is intended to built up concepts inductively by using concept forming operators, called *constructors*. *Concept names* are denoted by C and D. Atomic concepts, corresponding to unary predicate symbols, will

Constructor	Syntax	Logical correspondent
Conjunction	$C \sqcap D$	$C(x) \wedge_{\mathsf{min}} D(x)$
Disjunction	$C \sqcup D$	$C(x) \vee_{\mathsf{max}} D(x)$
Manipulator	$M_i C$	$M_i C(x)$
Value restr.	$\forall r.C$	$\forall_{Inf}\ y : \neg r(x,y) \vee_{\mathsf{max}} C(y)$
Existential restr.	$\exists r.C$	$\exists_{Sup}\ y : r(x,y) \wedge_{\mathsf{min}} C(y)$

Table 1: The language $\mathcal{ALC}_{F_M}$.

serve as a starting point to form more complex concepts. Further, *role names* are denoted by R and correspond to binary predicate symbols.

A crucial property of vague terminology is the use of modifications applied on verbs and nouns like "mostly", "more or less" or "very". The research area of fuzzy logic encompasses so-called linguistic hedges (see [12]) which are mathematical descriptions of modifications. In this paper, a subtype of the general notion is introduced, called (membership) manipulator. The manipulator is a triangular function from [0..1] to [0..1].

The supported language is called $\mathcal{ALC}_{F_M}$ and consists of the constructors seen in table 1. The meaning of the index F_M is that the language deals with infinitely many truth-values (Fuzzy) and a set of manipulators.

Definition 5 (Membership Manipulator) *A* (membership) manipulator *is a surjective and continuous function* $m : [0..1] \to [0..1]$ *such that there exist exactly one* $x_0 \in [0..1]$ *and exactly one* $x_1 \in [0..1]$ *as well as* $c_1 > 0$, *and* $c_2 < 0$ *with*

1. $m(x_0) = 0$ *and* $m(x_1) = 1$,
2. *if* $x_1 > 0$, *then* $m'(x) = c_1$ *for all* $x \in [0 \ldots x_1]$[1],
3. *if* $x_1 < 1$, *then* $m'(x) = c_2$ *for all* $x \in [x_1 \ldots 1]$.

Note, that in general the reverse relation yields a relation instead of a function. Nevertheless, the structure of the manipulator allows the decomposition of the reverse relation in at most two functions m_1^{-1}, m_2^{-1}, since m can be decomposed into two functions m_1, m_2 with

$$m(x) = \begin{cases} m_1(x) & \text{for } x \in [0..x_1] \\ m_2(x) & \text{for } x \in [x_1..1] \end{cases}$$

The domain of m_1^{-1} is $[m(0)..m(x_1)]$ and the domain of m_2^{-1} is $[m(1)..m(x_1)]$.

Figure 1 shows a realization of the manipulator MORE_OR_LESS. Since a membership degree of 1 does not completely match the semantics of this manipulator (if something is totally enlarged, it is not really completely more or less enlarged), the corresponding truth value is 0.8.

[1] The function m' denotes the derivative $\delta m / \delta x$ of m.

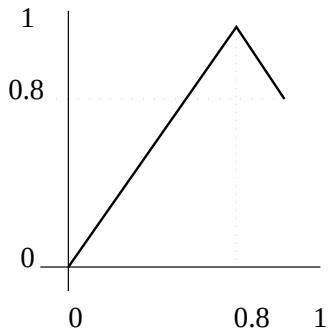

Figure 1: The membership manipulator MORE_OR_LESS

For two $\mathcal{ALC}_{F_M}$-concepts C and D it is interesting to know to which degree one concept is a special case of the other one. The procedure that computes the degree is called *subsumption algorithm*. As an example, consider the field of medicine where vague notions are nothing unusual. It might be interesting to know, to which degree the concept

$\texttt{hepatit\_liver} := \texttt{liver} \sqcap \ldots \sqcap \exists \texttt{sub\_organ.more\_or\_less(enlarged)}$

is a special case of the concept

$\texttt{enlarged\_liver} := \texttt{liver} \sqcap \ldots \sqcap \exists \texttt{sub\_organ.enlarged}.$

The degree of subsumption between two $\mathcal{ALC}_{F_M}$-concepts is defined by $C \sqsubseteq_{t_D} D$ if and only if $C' \models [D', t_D]$ where C', D' denote the logical corresponding expressions for C, D inductively defined by Table 1. The next section introduces an appropriate subsumption algorithm.

4 The Reasoning Algorithm and its Properties

The algorithm to compute the degree of subsumption between two concepts works on some special data structure which is introduced in the following.

Therefore, let V be a set of infinite many variables x_i. Let C be an arbitrary $\mathcal{ALC}_{F_M}$-concept and R an arbitrary role name.

Let $\mathcal{T}$ be the set of truth value designators satisfying the form $t_{x_i:C}$ resp. $t_{x_{i_1}x_{i_2}:R}$. Let further be $t_\top$ in $\mathcal{T}$. A mapping τ from $\mathcal{T}$ to [0..1] such that $\tau(t_\top) = 1$ is called *evaluation*.

Let variables $x_1, x_2 \in V$, a role name R and an $\mathcal{ALC}_{F_M}$-concept C. $[x_1 : C, t_{x_1:C}]$ and $[x_1 x_2 : r, t_{x_1 x_2:r}]$ are called *weighted assertions*. An *inequation* is of the form $\mathbf{op_1}(t_1)\,\mathbf{R}\,\mathbf{op_2}(t_2)$ or $\mathbf{op_1}(t_1)\,\mathbf{R}\,\mathbf{f}(\mathbf{op_2}(t_2), \mathbf{op_3}(t_3))$ with $t_i \in \mathcal{T}$, $\mathbf{R} \in \{=, \leq, \geq\}$, $\mathbf{op}_j \in \{id, m_i\}$, and $f \in \{\mathsf{min}, \mathsf{max}\}$. The operator id is defined as $id(t) = t$ for all $t \in \mathcal{T}$ and m_i denotes a manipulator.

Definition 6 (Extended ABox, Model and Consistency)

1. *An* extended ABox $\mathcal{E}$ *is a pair* $(\mathcal{A}, \mathcal{S})$*, where* $\mathcal{A}$ *is a finite set of weighted assertions and* $\mathcal{S}$ *is a finite set of inequations.*

2. *A pair* $(\mathfrak{A}, \tau)$ *is called* model *of an extended ABox* $\mathcal{E}$*, if and only if* $\mathfrak{A}$ *is an interpretation and* τ *is an evaluation, such that the following conditions are satisfied:*

   (a) *Exists an* $\mathfrak{A}$*-allocation* σ *such that for each* $[x : C, t_{x:C}] \in \mathcal{A}$: $\sigma, \mathfrak{A} \models [C(x), \tau(t_{x:C})]$ *and for each* $[x_1 x_2 : r, t_{x_1 x_2 : r}] \in \mathcal{A} : \sigma, \mathfrak{A} \models [r(x_1, x_2), \tau(t_{x_1 x_2 : r})]$.

   (b) *Let* $\mathsf{VAR}(\mathcal{S})$ *be the set of occurring variables in* $\mathcal{S}$*.*
   $\tau : \mathsf{VAR}(S) \to [0..1]$ *is a solution of* $\mathcal{S}$*, i.e. if each variable* t *in* $\mathcal{S}$ *is mapped to* $\tau(t) \in [0..1]$*, all inequations in* $\mathcal{S}$ *are satisfied.*

3. *An extended ABox* $\mathcal{E}$ *is called* $\tau(\mathcal{S})$*-consistent, if there exists a model* $(\mathfrak{A}, \tau)$ *of* $\mathcal{E}$*.*

The subsumption algorithm 10 computes the degree of subsumption that corresponds to the notion of semantic consequence (see Definition 4). The idea behind is as follows: The problem of subsumption is reduced to the problem of determining an adequate evaluation τ for an extended ABox that yields a solution for the system of inequations $\mathcal{S}$. The degree of subsumption is then determined as the minimum of all values obtained for some specific variable. To guarantee the soundness of the reduction step, it has to be shown that for all possible $\mathfrak{A}$-evaluations σ of an arbitrary interpretation $\mathfrak{A}$ that entails C to degree 1 and D to some arbitrary degree, a counterpart in form of a model of an extended ABox $\mathcal{E}$ can be found. This is done by the following lemma.

Lemma 7 (Satisfiability and Consistency)
For all values $t_D \in [0..1]$*, the set* $\{[C, 1], [D, t_D]\}$ *is satisfiable iff the extended ABox* $\mathcal{E}_0 = \{\{[x_0 : C, t_{x_0:C}], [x_0 : D, t_{x_0:D}]\}, \{t_{x_0:C} = t_\top\}\}$ *is* $\tau(\mathcal{S}_0)$*-consistent with* $\tau(t_\top) = 1$ *and* $\tau(t_{x_0:D}) = t_D$ *.*

Proof: Let σ be an $\mathfrak{A}$-allocation with $\sigma, \mathfrak{A} \models [C, 1]$ and $\sigma, \mathfrak{A} \models [D, t_D]$. Define τ as follows: $\tau(t_{x_0:C}) = 1, \tau(t_{x_0:D}) = t_D$. Since $t_{x_0:C} = t_\top$ is the only inequation in $\mathcal{S}_0$ and $\tau(t_{x_0:C}) = 1 = \tau(t_\top)$ is a solution of this inequation, obviously all inequations in $\mathcal{S}_0$ are satisfied. Then, $(\mathfrak{A}, \tau)$ is a model of $\mathcal{E}_0 = \{\{[x_0 : C, t_{x_0:C}], [x_0 : D, t_{x_0:D}]\}, \{t_{x_0:C} = t_\top\}\}$ and $\mathcal{E}_0$ is $\tau(\mathcal{S}_0)$-consistent.

Conversely, let $\{[x_0 : C, t_{x_0:C}], [x_0 : D, t_{x_0:D}]\}$ be $\tau(\mathcal{S}_0)$-consistent. Because of Definition 6, there exists a model $(\mathfrak{A}, \tau)$ with $\mathfrak{A}$-allocation σ such that $\sigma, \mathfrak{A} \models [C, \tau(t_{x_0:C})]$ and $\sigma, \mathfrak{A} \models [D, \tau(t_{x_0:D})]$ with $\tau(t_{x_0:C}) = \tau(t_\top) = 1$ and $\tau(t_{x_0:D}) = t_D$. Then, $\sigma, \mathfrak{A} \models [C, 1]$ and $\sigma, \mathfrak{A} \models [D, t_D]$. ∎

We are now ready to prove the main theorem of the paper.

Theorem 8 (Computation of the subsumption degree)
For two given $\mathcal{ALC}_{F_M}$-concepts C, D the degree of subsumption $C \sqsubseteq_{t_D} D$ is computable.

Theorem 8 is proved by introducing an algorithm that computes the degree of subsumption between two $\mathcal{ALC}_{F_M}$-concepts C, D. The algorithm is similar to the classic tableau method and hence based on the following completion rules.

Definition 9 (Completion Rules)

1. **Conjunction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\sqcap} (\mathcal{A}', \mathcal{S}')$
   if the following preconditions are satisfied:
   $[x : C_1 \sqcap C_2, t_{x:C_1 \sqcap C_2}] \in \mathcal{A}$ *and*
   $t_{x:C_1 \sqcap C_2} = \mathsf{min}(t_{x:C_1}, t_{x:C_2})$ *not in* $\mathcal{S}$
   then propagate:
   $\mathcal{A}' = \mathcal{A} \cup \{[x : C_1, t_{x:C_1}], [x : C_2, t_{x:C_2}]\}$ *and*
   $\mathcal{S}' = \mathcal{S} \cup \{t_{x:C_1 \sqcap C_2} = \mathsf{min}(t_{x:C_1}, t_{x:C_2})\}$.
2. **Disjunction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\sqcup} (\mathcal{A}', \mathcal{S}')$
   if the following preconditions are satisfied:
   $[x : C_1 \sqcup C_2, t_{x:C_1 \sqcup C_2}] \in \mathcal{A}$ *and*
   $t_{x:C_1 \sqcup C_2} = \mathsf{max}(t_{x:C_1}, t_{x:C_2})$ *not in* $\mathcal{S}$
   then propagate:
   $\mathcal{A}' = \mathcal{A} \cup \{[x : C_1, t_{x:C_1}], [x : C_2, t_{x:C_2}]\}$ *and*
   $\mathcal{S}' = \mathcal{S} \cup \{t_{x:C_1 \sqcup C_2} = \mathsf{max}(t_{x:C_1}, t_{x:C_2})\}$.
3. **Manipulator:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{M} (\mathcal{A}', \mathcal{S}'), (\mathcal{A}'', \mathcal{S}'')$
   if the following preconditions are satisfied:
   $[x : M(C), t_{x:M(C)}] \in \mathcal{A}$ *and*
   $t_{x:C} = m_1^{-1}(t_{x:M(C)}) \notin \mathcal{S}$ *and also*
   $t_{x:C} = m_2^{-1}(t_{x:M(C)}) \notin \mathcal{S}$
   then propagate:
   $\mathcal{A}' = \mathcal{A}'' = \mathcal{A} \cup \{[x : C, t_{x:C}]\}$ *and*
   $\mathcal{S}' = \mathcal{S} \cup \{t_{x:C} = m_1^{-1}(t_{x:M(C)}), t_{x:C} \leq x_1\}$ *and*
   $\mathcal{S}'' = \mathcal{S} \cup \{t_{x:C} = m_2^{-1}(t_{x:M(C)}), t_{x:C} \geq x_1\}$.
4. **Existential Restriction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\exists_1} (\mathcal{A}', \mathcal{S}')$
   if the following preconditions are satisfied:
   $[x : \exists r.C, t_{x:\exists r.C}] \in \mathcal{A}$ *and there exists no* y *for which*
   $t_{x:\exists r.C} = \mathsf{min}(t_{xy:r}, t_{y:C})$ *is in* $\mathcal{S}$
   then propagate for a new variable y*:*
   $\mathcal{A}' = \mathcal{A} \cup \{[xy : r, t_{xy:r}], [y : C, t_{y:C}]\}$ *and*
   $\mathcal{S}' = \mathcal{S} \cup \{t_{x:\exists r.C} = \mathsf{min}(t_{xy:r}, t_{y:C})\}$.

5. **Supremum Restriction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\exists_2} (\mathcal{A}', \mathcal{S}')$
if the following preconditions are satisfied:
$\{[x : \exists r.C, t_{x:\exists r.C}], [xy : r, t_{xy:r}]\} \subseteq \mathcal{A}$ *and for* y
$\{t_{x:\exists r.C} = \mathsf{min}(t_{xy:r}, t_{y:C}),$
$t_{x:\exists r.C} \geq \mathsf{min}(t_{xy:r}, t_{y:C})\}$ *is not in* $\mathcal{S}$ *then propagate:*
$\mathcal{A}' = \mathcal{A} \cup \{[y : C, t_{y:C}]\}$ *and*
$\mathcal{S}' = \mathcal{S} \cup \{t_{x:\exists r.C} \geq \mathsf{min}(t_{xy:r}, t_{y:C})\}$.

6. **Value Restriction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\forall_1} (\mathcal{A}', \mathcal{S}')$
if the following preconditions are satisfied:
$[x : \forall r.C, t_{x:\forall r.C}] \in \mathcal{A}$ *and there exists no* y *for which*
$t_{x:\forall r.C} = \mathsf{max}(1 - t_{xy:r}, t_{y:C})$ *is in* $\mathcal{S}$
then propagate for a new variable y*:*
$\mathcal{A}' = \mathcal{A} \cup \{[xy : r, t_{xy:r}], [y : C, t_{y:C}]\}$ *and*
$\mathcal{S}' = \mathcal{S} \cup \{t_{x:\forall r.C} = \mathsf{max}(1 - t_{xy:r}, t_{y:C})\}$.

7. **Infimum Restriction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\forall_2} (\mathcal{A}', \mathcal{S}')$
if the following preconditions are satisfied:
$\{[x : \forall r.C, t_{x:\forall r.C}], [xy : r, t_{xy:r}]\} \subseteq \mathcal{A}$ *and for* y
$\{\tau(t_{x:\forall r.C}) = \mathsf{max}(1 - t_{xy:r}, t_{y:C}),$
$t_{x:\forall r.C} \leq \mathsf{max}(1 - t_{xy:r}, t_{y:C})\}$ *is not in* $\mathcal{S}$ *then propagate:*
$\mathcal{A}' = \mathcal{A} \cup \{[y : C, t_{y:C}]\}$ *and*
$\mathcal{S}' = \mathcal{S} \cup \{t_{x:\forall r.C} \leq \mathsf{max}(1 - t_{xy:r}, t_{y:C})\}$.

To formulate the algorithm and some of its properties that are required to prove Theorem 8 the following notions will be needed.

An extended ABox $\mathcal{E}$ contains a *clash* if and only if there is no solution for the system of inequations $\mathcal{S}$. Otherwise, $\mathcal{E}$ is called *clash-free*.

An extended ABox $\mathcal{E}$ is called *complete* if and only if there is no further rule applicable to $\mathcal{E}$.

Algorithm 10 (t_D-minimal consistency algorithm for $\mathcal{ALC}_{F_M}$)
The *subsumption algorithm* gets as inputs two $\mathcal{ALC}_{F_M}$-concepts C, D and computes the degree of subsumption $C \sqsubseteq_{t_D} D$. It works on a tree where each node is labeled with an extended ABox. It starts with the tree only consisting of the root labeled with $\mathcal{E}_0 = \{\mathcal{A}_0, \mathcal{S}_0\}$ where $\mathcal{A}_0 = \{[x_0 : C, t_{x_0:C}], [x_0 : D, t_{x_0:D}]\}$ and $\mathcal{S}_0 = \{t_{x_0:C} = t_\top\}$. Apply the completion rules of Definition 9 to leaves until all leaves are labeled with complete extended ABoxes. Applying a rule $\mathcal{E} \to \mathcal{E}_i, 1 \leq i \leq 2$, to such a leaf leads to the creation of at most two new successors of this node, each labeled with one of the extended ABoxes $\mathcal{E}_i$. Then, an optimization method is applied to all complete extended ABoxes to solve their systems of inequations. If all systems of inequations are not solvable, the algorithm answers "1". Otherwise, it answers with the minimum of all computed values $\tau(t_{x_0:D})$. $\diamond$

Remark 11 *$\mathcal{ALC}$ (originated in [10]) is – roughly spoken – the crisp base for the language $\mathcal{ALC}_{F_M}$ without manipulators and arbitrary truth values in* [0..1] *for concepts. A tableau algorithm for $\mathcal{ALC}$ usually encompasses only a single rule for $\exists$ and a single one for $\forall$. Since $\exists$ in $\mathcal{ALC}_{F_M}$ is based on the* Supremum *and $\forall$ on the* Infimum*, it has to be guaranteed that these values will be obtained from fixed variables x, y where y is a so-called role successor of x, i.e. exists R such that $[xy : r, t_{xy:r}] \in \mathcal{A}$ (Existential and Value Restriction rules). Further, it is imperative that all values that are obtained from other role successors of x are smaller than the* Supremum *resp. larger than the* Infimum *(Supremum and Infimum Restriction rules).*

Each complete extended ABox encompasses a system of inequations $\mathcal{S}$. For each $\mathcal{S}$, the lowest possible value $\tau(t_{x_0:D})$ is determined. To obtain these minimal values for all complete extended ABoxes, an appropriate procedure from the domain of optimizing methods was selected (for more details, see the proof of Lemma 14).

Another important notion is the *extension* of an evaluation:

Definition 12 (Extension) *Given two systems of inequations $\mathcal{S}, \mathcal{S}'$ with $\mathcal{S} \subseteq \mathcal{S}'$. The evaluation $\tau' : \mathsf{VAR}(\mathcal{S}') \to [0..1]$ is called* extension *of the evaluation $\tau : \mathsf{VAR}(\mathcal{S}) \to [0..1]$ iff $\forall t_x \in \mathsf{VAR}(\mathcal{S}) : \tau'(t_x) = \tau(t_x)$.*

Lemma 13 *1. Each completion rule $\longrightarrow$ that can be applied to $\mathcal{E}$ satisfies: $\mathcal{E}$ is $\tau(\mathcal{S})$-consistent if and only if there is an $\mathcal{E}'$ with $\mathcal{E} \longrightarrow \mathcal{E}'$ and $\mathcal{E}'$ is $\tau'(\mathcal{S}')$-consistent where τ' is an expansion of τ.*

2. A clash-free and complete extended ABox $\mathcal{E}$ is $\tau(\mathcal{S})$-consistent.

3. An extended ABox $\mathcal{E}$ that contains a clash is not $\tau(S)$-consistent for an arbitrary τ.

4. The t_D-minimal consistency algorithm terminates.

Proof: We prove (1) by considering each completion rule.

1. **Conjunction:** Because of $\mathcal{A} \subseteq \mathcal{A}'$ and $\mathcal{S} \subseteq \mathcal{S}'$ each model of $\mathcal{E}'$ is also a model of $\mathcal{E}$.

   Conversely, let $(\mathfrak{A}, \tau)$ be a model of $\mathcal{E}$ and $\longrightarrow_\sqcap$ has been applied on $[x : C_1 \sqcap C_2, t_{x:C_1 \sqcap C_2}]$. We define an extension τ' of τ such that $(\mathfrak{A}, \tau')$ yields a model of $\mathcal{E}' = (\mathcal{A}', \mathcal{S}')$ where $\mathcal{A}' = \mathcal{A} \cup \{[x : C_1, t_{x:C_1}], [x : C_2, t_{x:C_2}]\}$ and $\mathcal{S}' = \mathcal{S} \cup \{t_{x:C_1 \sqcap C_2} = \mathsf{min}(t_{x:C_1}, t_{x:C_2})\}$. Since $(\mathfrak{A}, \tau)$ is a model of $\mathcal{E}$, there exists an $\mathfrak{A}$-allocation σ such that $\sigma, \mathfrak{A} \models [C_1(x) \wedge C_2(x), \tau(t_{x:C_1 \sqcap C_2})]$ and because of Definition 3, there are values $t_{C_1}, t_{C_2} \in [0..1]$ such that $\sigma, \mathfrak{A} \models [C_1(x), t_{C_1}]$ and $\sigma, \mathfrak{A} \models [C_2(x), t_{C_2}]$ and further, $\tau(t_{x:C_1 \sqcap C_2}) = \mathsf{min}(t_{C_1}, t_{C_2})$. Then, a solution of $\mathcal{S}'$ is obtained by extending τ to τ' by $\tau'(t_{x:C_1}) = t_{C_1}, \tau'(t_{x:C_2}) = t_{C_2}$. Hence, $\mathcal{E}'$ is $\tau'(\mathcal{S}')$-consistent.

2. **Disjunction:** Because of $\mathcal{A} \subseteq \mathcal{A}'$ and $\mathcal{S} \subseteq \mathcal{S}'$ each model of $\mathcal{E}'$ is also a model of $\mathcal{E}$.

   Conversely, let $(\mathfrak{A}, \tau)$ be a model of $\mathcal{E}$ and $\longrightarrow_{\sqcup}$ has been applied on $[x : C_1 \sqcup C_2, t_{x:C_1 \sqcup C_2}]$. We define an extension τ' of τ such that $(\mathfrak{A}, \tau')$ yields a model of $\mathcal{E}' = (\mathcal{A}', \mathcal{S}')$ where $\mathcal{A}' = \mathcal{A} \cup \{[x : C_1, t_{x:C_1}], [x : C_2, t_{x:C_2}]\}$ and $\mathcal{S}' = \mathcal{S} \cup \{t_{x:C_1 \sqcup C_2} = \mathsf{max}(t_{x:C_1}, t_{x:C_2})\}$. Since $(\mathcal{A}, \tau)$ is a model of $\mathcal{E}$, there exists an $\mathfrak{A}$-allocation σ such that $\sigma, \mathfrak{A} \models [C_1(x) \vee C_2(x), \tau(t_{x:C_1 \sqcup C_2})]$. By Definition 3, we get values $t_{C_1}, t_{C_2} \in [0..1]$ such that $\sigma, \mathfrak{A} \models [C_1(x), t_{C_1}], \sigma, \mathfrak{A} \models [C_2(x), t_{C_2}]$ and $\tau(t_{x:C_1 \sqcup C_2}) = \mathsf{max}(t_{C_1}, t_{C_2})$. Then, a solution of $\mathcal{S}'$ is obtained by extending τ to τ' by $\tau'(t_{x:C_1}) = t_{C_1}, \tau'(t_{x:C_2}) = t_{C_2}$. Hence, $\mathcal{E}'$ is $\tau'(\mathcal{S}')$-consistent.

3. **Manipulator:** Because of $\mathcal{A} \subseteq \mathcal{A}'$ and $\mathcal{S} \subseteq \mathcal{S}'$ as well as $\mathcal{S} \subseteq \mathcal{S}''$, each model of $\mathcal{E}'$ and $\mathcal{E}''$, respectively, is also a model of $\mathcal{E}$.

   Conversely, let $(\mathfrak{A}, \tau)$ be a model of $\mathcal{E}$ and $\longrightarrow_M$ has been applied on $[x : M(C),$
   $t_{x:M(C)}]$. We show that there exists an extension τ' of τ such that $(\mathfrak{A}, \tau)$ is a model of $\mathcal{E}'$ or $\mathcal{E}''$.

   Since $(\mathfrak{A}, \tau)$ is a model of $\mathcal{E}$, there exists an $\mathfrak{A}$-allocation σ such that $\sigma, \mathfrak{A} \models [M(C(x)), \tau(t_{x:M(C)})]$. By Definition 3, we get $t_C \in [0..1]$ such that $\sigma, \mathfrak{A} \models [C(x), t_C]$ and $m(t_C) = \tau(t_{x:M(C)})$. According to Definition 5, for each manipulator m, there exists a unique value $x_1 \in [0..1]$ with $m(x_1) = 1$. Further, at least one of the following two cases holds:

   (1) $t_C \in [0 \ldots x_1]$ and $m(t_c) = m_1(t_C)$, or
   (2) $t_C \in [x_1 \ldots 1]$ and $m(t_C) = m_2(t_C)$.

   Define the extension τ' of τ by $\tau'(t_{x:C} := t_C$. If case (1) is satisfied, then $(\mathfrak{A}, \tau')$ is a model of $\mathcal{E}'$. Otherwise, $(\mathfrak{A}, \tau')$ is a model of $\mathcal{E}''$. Hence, $\mathcal{E}'$ is $\tau(\mathcal{S}')$-consistent or $\mathcal{E}''$ is $\tau'(\mathcal{S}'')$-consistent.

4. **Existential Restriction:** Because of $\mathcal{A} \subseteq \mathcal{A}'$ and $\mathcal{S} \subseteq \mathcal{S}'$, each model of $\mathcal{E}'$ is also model of $\mathcal{E}$.

   Conversely, let $(\mathfrak{A}, \tau)$ be a model of $\mathcal{E}$ and $\longrightarrow_{\exists_1}$ has been applied on $[x : \exists r.C,$
   $t_{x:\exists r.C}]$. We show that there exists an extension τ' of τ such that $(\mathfrak{A}, \tau')$ yields a model of $\mathcal{E}' = (\mathcal{A}', \mathcal{S}')$ where y is the new variable in $\mathcal{E}'$, and $\mathcal{A}' = \mathcal{A} \cup \{[xy : r, t_{xy:r}], [y : C, t_{y:C}]\}$ and $\mathcal{S}' = \mathcal{S} \cup \{t_{x:\exists r.C}) = \mathsf{min}(t_{xy:r}, t_{y:C})\}\}$.

   Since $(\mathfrak{A}, \tau)$ is model of $\mathcal{E}$, the expression $[x : \exists r.C]$ is valid to the degree $\tau(t_{x:\exists r.C})$. Thus, there exists an $\mathfrak{A}$-allocation σ and an individual $\xi \in \Delta$ such that, for the new variable y, we have $\sigma\langle y := \xi\rangle \models [r(x, y), t_r]$ and $\sigma\langle y := \xi\rangle \models [C(y), t_C]$ and $\mathsf{min}(t_r, t_C) = \tau(t_{x:\exists r.C})$.

   Let $\sigma' := \sigma\langle y := \xi\rangle$ and extend τ to τ' by $\tau'(t_{xy:r}) := t_r$ and $\tau'(t_{y:C}) := t_C$.
   Since $\sigma', \mathfrak{A} \models [r(x, y) \wedge_{min} C(y), \tau(t_{x:\exists r.C})]$ and $\tau'(t_{x:\exists r.C}) = \mathsf{min}(\tau'(t_{xy:r}), \tau'(t_{y:C}))$, we get that $(\mathfrak{A}, \tau')$ is a model of $\mathcal{E}'$ and $\mathcal{E}'$ is $\tau'(\mathcal{S}')$-consistent.

5. **Supremum Restriction:** Because of $\mathcal{A} \subseteq \mathcal{A}'$ and $\mathcal{S} \subseteq \mathcal{S}'$, each model of $\mathcal{E}'$ is also model of $\mathcal{E}$.

   Conversely, let $(\mathfrak{A}, \tau)$ be a model of $\mathcal{E}$ and $\longrightarrow_{\exists_2}$ has been applied on $\{[x : \exists r.C, t_{x:\exists r.C}], [xy : r, t_{xy:r}]\}$. We define an extension τ' of τ such that $(\mathfrak{A}, \tau')$ is a model of $\mathcal{E}' = (\mathcal{A}', \mathcal{S}')$ where $\mathcal{A}' = \mathcal{A} \cup \{[y : C, t_{xy:C}]\}$ and $\mathcal{S}' = \mathcal{S} \cup \{t_{x:\exists r.C}) \geq \mathsf{min}(t_{xy:r}, t_{y:C})\}\}$.

   Since $(\mathfrak{A}, \tau)$ is model of $\mathcal{E}$, there exists an $\mathfrak{A}$-allocation σ such that $\sigma, \mathfrak{A} \models [x : \exists r.C, \tau(t_{x:\exists r.C})$. By the semantics of $\mathcal{ALC}_{F_M}$ (see Table 1) we get $\sigma, \mathfrak{A} \models [\exists_{Sup} z.r(x, z) \wedge_{min} C(z), \tau(t_{x:\exists r.C}]$, i.e., for each r-successor ξ of $\sigma(x)$ in $\mathfrak{A}$ we have $\tau(t_{x:\exists r.C}) \geq \mathsf{min}(t_1, t_2)$ where t_1, t_2 are determined by $\sigma\langle z := \xi\rangle, \mathfrak{A} \models [r(x, z), t_1]$ and $\sigma\langle z := \xi\rangle, \tau \models [C(z), t_2]$. Since $(\mathfrak{A}, \tau)$ is a model of $\mathcal{E}$, $[xy : r, t_{xy:r}] \in \mathcal{A}$ implies that $\sigma(y)$ is an r-successor of $\sigma(x)$ in $\mathfrak{A}$. It follows $\tau(t_{x:\exists r.C}) \geq \mathsf{min}(t_r, t_C)$ where $\sigma, \mathfrak{A} \models [r(x, y), t_r]$ and $\sigma, \mathfrak{A} \models [C(y), t_C]$.

   Define the extension τ' of τ by $\tau'(t_{y:C} := t_C$. Then, $(\mathfrak{A}, \tau')$ yields a model of $\mathcal{E}'$ and $\mathcal{E}'$ is $\tau(\mathcal{S}')$-consistent.

6. **Value Restriction:** Because of $\mathcal{A} \subseteq \mathcal{A}'$ and $\mathcal{S} \subseteq \mathcal{S}'$, each model of $\mathcal{E}'$ is also model of $\mathcal{E}$.

   Conversely, let $(\mathfrak{A}, \tau)$ be a model of $\mathcal{E}$ and *rall* has been applied on $[x : \forall r.C, t_{x:\exists r.C}]$. We show that there exists an extension τ' of τ such that $(\mathfrak{A}, \tau')$ yields a model of $\mathcal{E}' = (\mathcal{A}', \mathcal{S}')$ where y is the new variable in $\mathcal{E}'$, and $\mathcal{A}' = \mathcal{A} \cup \{[xy : r, t_{xy:r}], [y : C, t_{y:C}]\}$ and $\mathcal{S}' = \mathcal{S} \cup \{t_{x:\forall r.C}) = \mathsf{max}(1 - t_{xy:r}, t_{y:C})\}\}$.

   Since $(\mathfrak{A}, \tau)$ is model of $\mathcal{E}$, the expression $[x : \forall r.C]$ is valid to the degree $\tau(t_{x:\forall r.C})$. Thus, there exists an $\mathfrak{A}$-allocation σ and an individual $\xi \in \Delta$ such that, for the new variable y, we have $\sigma\langle y := \xi\rangle \models [r(x, y), t_r]$, i.e., $\sigma\langle y := \xi\rangle \models [\neg r(x, y), 1 - t_r]$, and $\sigma\langle y := \xi\rangle \models [C(y), t_C]$ and $\mathsf{max}(1 - t_r, t_C) = \tau(t_{x:\forall r.C})$.

   Let $\sigma' := \sigma\langle y := \xi\rangle$ and extend τ to τ' by $\tau'(t_{xy:r}) := t_r$ and $\tau'(t_{y:C}) := t_C$. Since $\sigma', \mathfrak{A} \models [\neg r(x, y) \vee_{max} C(y), \tau(t_{x:\exists r.C})]$ and $\tau'(t_{x:\forall r.C}) = \mathsf{max}(1 - \tau'(t_{xy:r}), \tau'(t_{y:C}))$, we get that $(\mathfrak{A}, \tau')$ is a model of $\mathcal{E}'$ and $\mathcal{E}'$ is $\tau'(\mathcal{S}')$-consistent.

7. **Infimum Restriction:** Because of $\mathcal{A} \subseteq \mathcal{A}'$ and $\mathcal{S} \subseteq \mathcal{S}'$, each model of $\mathcal{E}'$ is also model of $\mathcal{E}$.

   Conversely, let $(\mathfrak{A}, \tau)$ be a model of $\mathcal{E}$ and *ralli* has been applied on $\{[x : \forall r.C, t_{x:\forall r.C}], [xy : r, t_{xy:r}]\}$. We define an extension τ' of τ such that $(\mathfrak{A}, \tau')$ is a model of $\mathcal{E}' = (\mathcal{A}', \mathcal{S}')$ where $\mathcal{A}' = \mathcal{A} \cup \{[y : C, t_{xy:C}]\}$ and $\mathcal{S}' = \mathcal{S} \cup \{t_{x:\forall r.C}) \geq \mathsf{max}(1 - t_{xy:r}, t_{y:C})\}\}$.

   Since $(\mathfrak{A}, \tau)$ is model of $\mathcal{E}$, there exists an $\mathfrak{A}$-allocation σ such that $\sigma, \mathfrak{A} \models [x : \forall r.C, \tau(t_{x:\forall r.C})$. By the semantics of $\mathcal{ALC}_{F_M}$ (see Table 1) we get $\sigma, \mathfrak{A} \models [\forall_{Inf} z.r(x, z) \vee_{max} C(z), \tau(t_{x:\forall r.C}]$, i.e., for each r-successor ξ of $\sigma(x)$ in $\mathfrak{A}$ we have $\tau(t_{x:\forall r.C}) \leq \mathsf{max}(1 - t_1, t_2)$ where t_1, t_2 are determined

by $\sigma\langle z := \xi\rangle, \mathfrak{A} \models [r(x,z), t_1]$ and $\sigma\langle z := \xi\rangle, \tau \models [C(z), t_2]$. Since $(\mathfrak{A}, \tau)$ is a model of $\mathcal{E}$, $[xy : r, t_{xy:r}] \in \mathcal{A}$ implies that $\sigma(y)$ is an r-successor of $\sigma(x)$ in $\mathfrak{A}$. It follows $\tau(t_{x:\forall r.C}) \leq \mathsf{max}(1 - t_r, t_C)$ where $\sigma, \mathfrak{A} \models [r(x,y), t_r]$ and $\sigma, \mathfrak{A} \models [C(y), t_C]$.

Define the extension τ' of τ by $\tau'(t_{y:C} := t_C$. Then, $(\mathfrak{A}, \tau')$ yields a model of $\mathcal{E}'$ and $\mathcal{E}'$ is $\tau(\mathcal{S}')$-consistent.

(2) Since $\mathcal{E}$ is clash-free, there exists an evaluation τ that is a solution of $\mathcal{S}$. Define an interpretation as follows:

- Δ is the set of variables in $\mathcal{A}$.
- For all role names R μ_R is defined as
  $\mu_r(x,y) := \begin{cases} \tau(t_{xy:r}) & \text{if } [xy : r, t_{xy:r}] \in \mathcal{A} \\ 0 & \text{otherwise} \end{cases}$
- For all concept names A μ_A is defined as
  $\mu_A(x) := \begin{cases} \tau(t_{x:A}) & \text{if } [x : A, t_{x:A}] \in \mathcal{A} \\ 0 & \text{otherwise} \end{cases}$

Let $\sigma := id$, i.e. $\sigma(x) := x$. We show that $(\mathfrak{A}, \tau)$ is a model of the complete extended ABox $\mathcal{E}$, i.e. $\sigma, \mathfrak{A}$ satisfies each weighted assertion in $\mathcal{A}$ and τ is a solution for $\mathcal{S}$. Proof by induction on the structure of concepts:

Base case:

For all weighted assertions of the form $[xy : r, t_{xy:r}]$ and $[x : C, t_{x:C}]$ the claim follows by definition of μ_R and μ_A.

Induction Step:

1. **Conjunction:** $[x : C_1 \sqcap C_2, t_{x:C_1 \sqcap C_2}] \in \mathcal{A}$. Since $\mathcal{E}$ is complete, $\longrightarrow_\sqcap$ is no longer applicable. Thus, we have $\{[x : C_1, t_{x:C_1}], [x : C_2, t_{x:C_2}]\} \subseteq \mathcal{A}$ and $t_{x:C_1 \sqcap C_2} = \mathsf{min}(t_{x:C_1}, t_{x:C_2}) \in \mathcal{S}$. By induction, $\sigma, \mathfrak{A} \models [x : C_1, \tau(t_{x:C_1})]$ and $\sigma, \mathfrak{A} \models [x : C_2, \tau(t_{x:C_2})]$. Since τ is a solution of $\mathcal{S}$, we have $t_{x:C_1 \sqcap C_2} = \mathsf{min}(t_{x:C_1}, t_{x:C_2})$ and hence, $\sigma, \tau \models [x : C_1 \sqcap C_2, t_{x:C_1 \sqcap C_2}]$.
2. **Disjunction:** $[x : C_1 \sqcup C_2, t_{x:C_1 \sqcup C_2}] \in \mathcal{A}$. Since $\mathcal{E}$ is complete, $\longrightarrow_\sqcup$ is no longer applicable and therefore $\{[x : C_1, t_{x:C_1}], [x : C_2, t_{x:C_2}]\} \subseteq \mathcal{A}$ and $t_{x:C_1 \sqcup C_2} = \mathsf{max}(t_{x:C_1}, t_{x:C_2}) \in \mathcal{S}$. By induction, $\sigma, \mathfrak{A} \models [x : C_1, \tau(t_{x:C_1})]$ and $\sigma, \mathfrak{A} \models [x : C_2, \tau(t_{x:C_2})]$. Since τ is a solution of $\mathcal{S}$, we have $t_{x:C_1 \sqcup C_2} = \mathsf{max}(t_{x:C_1}, t_{x:C_2})$ and hence, $\sigma, \mathfrak{A} \models [x : C_1 \sqcup C_2, t_{x:C_1 \sqcup C_2}]$.
3. **Manipulator:** $[x : M(C), t_{x:M(C)}] \in \mathcal{A}$. Since $\mathcal{E}$ is complete, $\longrightarrow_M$ is no longer applicable. Thus, one of the following two cases must hold:
   (1) $[x : C, t_{x:C}] \in \mathcal{A}$ and $\{t_{x:C} = m_1^{-1}(t_{x:M(C)}), t_{x:C} \leq x_1\} \subseteq \mathcal{S}$, or
   (2) $[x : C, t_{x:C}] \in \mathcal{A}$ and $\{t_{x:C} = m_2^{-1}(t_{x:M(C)}), t_{x:C} \geq x_1\} \subseteq \mathcal{S}$.

By induction, $\sigma, \mathfrak{A} \models [x : C, \tau(t_{x:C})]$. If the case (1) is true, then we have $0 \leq \tau(t_{x:C}) \leq x_1$ and hence $m(\tau(t_{x:C})) = m_1(\tau(t_{x:C}))$. So, $\tau(t_{x:C}) = m_1^{-1}(\tau(t_{x:M(C)}))$ implies that $\tau(t_{x:M(C)}) = m(\tau(t_{x:C}))$ and $\sigma, \mathfrak{A} \models [x : M(C), \tau(t_{x:M(C)})]$.

If the case (2) is true, then we have $x_1 \leq \tau(t_{x:C}) \leq 1$ and hence $m(\tau(t_{x:C})) = m_2(\tau(t_{x:C}))$. So, $\tau(t_{x:C}) = m_2^{-1}(\tau(t_{x:M(C)}))$ implies that $\tau(t_{x:M(C)}) = m(\tau(t_{x:C}))$ and $\sigma, \mathfrak{A} \models [x : M(C), \tau(t_{x:M(C)})]$.

4. **Existential Restriction:** $[x : \exists r.C, t_{x:\exists r.C}] \in \mathcal{A}$. Since $\mathcal{E}$ is complete, $\longrightarrow_{\exists_1}$ is not applicable on $\mathcal{E}$. Thus, there exists a variable y such that $\{[xy : r, t_{xy:r}], [y : C, t_{y:C}]\} \subseteq \mathcal{A}$ and $t_{x:\exists r.C} = \mathsf{min}(t_{xy:r}, t_{y:C}) \in \mathcal{S}$. By induction, $\sigma, \mathfrak{A} \models [r(x, y), \tau(t_{xy:r})]$ and $\sigma, \mathfrak{A} \models [C(y), \tau(t_{y:C})]$. Since τ is a solution of $\mathcal{S}$, we have $t_{x:\exists r.C} = \mathsf{min}(t_{xy:r}, t_{y:C})$.

   According to Definition 3 it remains to show that $\tau(t_{\exists r.C}) = sup\{t' \mid$ exists $\xi \in \Delta$ with $\sigma\langle y := \xi\rangle, \mathfrak{A} \models [\exists y.r(x, y) \wedge_{min} C(y), t']\}$. For the given interpretation $\mathfrak{A}$ and $\mathfrak{A}$-allocation σ, we must show that, for all r-successors z of x in $\mathcal{A}$, $\tau(t_{x:\exists r.C}) \geq \mathsf{min}(\tau(t_{xz:r}, \tau(t_{z:C}))$.

   Since $\mathcal{E}$ is complete, $\longrightarrow_{\exists_2}$ is not applicable on $\mathcal{E}$. Thus, for an arbitrary r-successor z of x in $\mathcal{A}$, we have $t_{x:\exists r.C} \geq \mathsf{min}(t_{xz:r}, t_{z:C} \in \mathcal{S}$. Since τ is a solution of $\mathcal{S}$, we get $\tau(t_{x:\exists r.C}) \geq \mathsf{min}(\tau(t_{xz:r}, \tau(t_{z:C}))$, and hence $\sigma, \mathfrak{A} \models [x : \exists r.C, \tau(t_{x:\exists r.C})]$.

5. **Value Restriction:** $[x : \forall r.C, t_{x:\forall r.C}] \in \mathcal{A}$. Since $\mathcal{E}$ is complete, *rall* is not applicable on $\mathcal{E}$. Thus, there exists a variable y such that $\{[xy : r, t_{xy:r}], [y : C, t_{y:C}]\} \subseteq \mathcal{A}$ and $t_{x:\forall r.C} = \mathsf{max}(1 - t_{xy:r}, t_{y:C}) \in \mathcal{S}$. By induction, $\sigma, \mathfrak{A} \models [r(x, y), \tau(t_{xy:r})]$ and $\sigma, \mathfrak{A} \models [C(y), \tau(t_{y:C})]$. Since τ is a solution of $\mathcal{S}$, we have $t_{x:\forall r.C} = \mathsf{max}(1 - t_{xy:r}, t_{y:C})$.

   According to Definition 3 it remains to show that $\tau(t_{\forall r.C}) = inf\{t' \mid$ exists $\xi \in \Delta$ with $\sigma\langle y := \xi\rangle, \mathfrak{A} \models [\exists y.\neg r(x, y) \vee_{max} C(y), t']\}$. For the given interpretation $\mathfrak{A}$ and $\mathfrak{A}$-allocation σ, we must show that, for all r-successors z of x in $\mathcal{A}$, $\tau(t_{x:\forall r.C}) \leq \mathsf{max}(1 - \tau(t_{xz:r}, \tau(t_{z:C}))$.

   Since $\mathcal{E}$ is complete, *ralli* is not applicable on $\mathcal{E}$. Thus, for an arbitrary r-successor z of x in $\mathcal{A}$, we have $t_{x:\forall r.C} \leq \mathsf{max}(1 - t_{xz:r}, t_{z:C} \in \mathcal{S}$. Since τ is a solution of $\mathcal{S}$, we get $\tau(t_{x:\forall r.C}) \leq \mathsf{max}(1 - \tau(t_{xz:r}, \tau(t_{z:C}))$, and hence $\sigma, \mathfrak{A} \models [x : \forall r.C, \tau(t_{x:\forall r.C})]$.

(3) Obvious.
(4) The termination proof is similar to the one for the tableau introduced in [1]. The idea is as follows: Define a mapping κ from extended ABoxes $(\mathcal{A}, \mathcal{S})$ to tuples of fixed length of nonnegative integers such that for each completion rule

$$(\mathcal{A}, \mathcal{S}) \longrightarrow (\mathcal{A}', \mathcal{S}') \implies \kappa(\mathcal{A}, \mathcal{S}) \succ \kappa(\mathcal{A}', \mathcal{S}')$$

is valid, where $\succ$ denotes the lexicographic ordering on tuples. Since the lexicographic ordering is well-founded, this implies termination of the algorithm.

In the following, only extended ABoxes $(\mathcal{A}, \mathcal{S})$ are considered that are obtained by applying the completion rules to $\{x_0 : [C, t_{x_0:C}], x_0 : [D, t_{x_0:D}]\}$. x_0 is denoted as the *root node* of the extended ABox.

The *and/or/manipulator-size* $|C|_{\sqcap/\sqcup/M}$ of a concept C is defined as the number of occurrences of conjunction, disjunction and manipulator constructors in C. Obviously, it is $|C_1 \sqcap C_2|_{\sqcap/\sqcup/M} = |C_1 \sqcup C_2|_{\sqcap/\sqcup/M} = |C_1|_{\sqcap/\sqcup/M} + |C_2|_{\sqcap/\sqcup/M} + 1$ and $|M(C)|_{\sqcap/\sqcup/M} = |C|_{\sqcap/\sqcup/M} + 1$.

The *maximal role depth* $\mathsf{depth}(C)$ of a concept C is defined as follows:

$$\begin{array}{lllll} \mathsf{depth}(A) & := & 0, & & \\ \mathsf{depth}(M(C)) & := & \mathsf{depth}(C), & & \\ \mathsf{depth}(C_1 \sqcap C_2) & := & \mathsf{depth}(C_1 \sqcup C_2) & := & \mathsf{max}\{\mathsf{depth}(C_1), \mathsf{depth}(C_2)\}, \\ \mathsf{depth}(\forall r.C) & := & \mathsf{depth}(\exists r.C) & := & 1 + \mathsf{depth}(C). \end{array}$$

Let m_0 be the maximum of the role depth of C and D. The following observations are a direct consequence of the definition of the completion rules. Each variable x in an extended ABox $(\mathcal{A}, \mathcal{S})$ is accessed from x_0 by a unique role chain $[x_0, x_1 : r_1, t_{x_0,x_1:r_1}], \ldots, [x_{n-1}, x : r_n, t_{x_{n-1},x:r_n}]$ in $\mathcal{A}$. Consequently, each variable x has a unique level $\mathsf{level}(x)$ in $\mathcal{A}$. The level of x_0 is defined as 0. If x is accessible from x_0 in $\mathcal{A}$ by a role chain of length m, then for each assertion $[x : C, t_{x:C}]$ in $\mathcal{A}$, the maximal role depth of C is bounded by $m_0 - m$. Therefore, $\mathsf{level}(x)$ is bounded by m_0 and the level of each variable in $\mathcal{A}$ is an integer between 0 and m_0.

Now, a mapping $\kappa(\mathcal{A}, \mathcal{S})$ from an extended ABox $(\mathcal{A}, \mathcal{S})$ to a $3(m_0+1)$-tuples of nonnegative integers is defined by

$$\kappa(\mathcal{A}, \mathcal{S}) := (\kappa_0, \ldots, \kappa_{m_0}),$$

where $\kappa_l := (k_{l1}, k_{l2}, k_{l3})$ and

- k_{l1} is the sum of the and/or/manipulator-sizes $|C|_{\sqcap/\sqcup/M}$ of all assertions $[x : C, t_{x:C}]$ in $\mathcal{A}$ such that $\mathsf{level}(x) = l$ and the conjunction, disjunction or manipulator rule is applicable to $[x : C, t_{x:C}]$ in $(\mathcal{A}, \mathcal{S})$.
- k_{l2} is the sum of the number of all assertions $[x : \exists r.C, t_{x:\exists r.C}]$ in $\mathcal{A}$ and the number of all assertions $[x : \forall r.C, t_{x:\forall r.C}]$ in $\mathcal{A}$ such that $\mathsf{level}(x) = l$ and the existential restriction rule (value restriction rule) is applicable to $[x : \exists r.C, t_{x:\exists r.C}]$ ($[x : \forall r.C, t_{x:\forall r.C}]$) in $(\mathcal{A}, \mathcal{S})$.
- k_{l3} is the sum of the number of all pairs of assertions $[x : \exists r.C, t_{x:\exists r.C}]$, $[xy : r, t_{xy:r}]$ and the number of all pairs of assertions $[x : \forall r.C, t_{x:\forall r.C}]$, $[xy : r, t_{xy:r}]$ in $\mathcal{A}$ such that $\mathsf{level}(x) = l$ and the supremum restriction rule (infimum restriction rule) is applicable to $[x : \exists r.C, t_{x:\exists r.C}]$, $[xy : r, t_{xy:r}]$ ($[x : \forall r.C, t_{x:\forall r.C}]$, $[xy : r, t_{xy:r}]$) in $(\mathcal{A}, \mathcal{S})$.

In the last part of the termination proof, for each completion rule it has to be shown that $(\mathcal{A}, \mathcal{S}) \longrightarrow (\mathcal{A}', \mathcal{S}')$ implies $\kappa(\mathcal{A}, \mathcal{S}) \succ \kappa(\mathcal{A}', \mathcal{S}')$.

1. Conjunction: $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\sqcap} (\mathcal{A}', \mathcal{S}')$

Assume that the rule is applied to the assertion $[x : C_1 \sqcap C_2, t_{x:C_1 \sqcap C_2}]$

in $\mathcal{A}$. Let $l := \mathsf{level}(x)$ and κ_l, κ'_l be the tuples associated with level l in $(\mathcal{A}, \mathcal{S})$ and $(\mathcal{A}', \mathcal{S}')$.

It is valid that $k_{l1} < k'_{l1}$, because in k'_{l1}, $|C_1 \sqcap C_2|_{\sqcap/\sqcup/M}$ is replaced by a number that is less or equal to $|C_1|_{\sqcap/\sqcup/M} + |C_2|_{\sqcap/\sqcup/M} < |C_1 \sqcap C_2|_{\sqcap/\sqcup/M}$. Since tuples are compared with the lexicographic ordering, an increase in the second or third component of κ'_l and κ'_m for $m > l$ is irrelevant. In a tuple κ'_m with $m < l$, none of the three components is changed. Therefore, $\kappa(\mathcal{A}, \mathcal{S}) \succ \kappa(\mathcal{A}', \mathcal{S}')$.

2. **Disjunction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\sqcup} (\mathcal{A}', \mathcal{S}')$
3. **Manipulator:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{M} (\mathcal{A}', \mathcal{S}')$
   Analogous to the conjunction rule.

4. **Existential Restriction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\exists_1} (\mathcal{A}', \mathcal{S}')$
   Assume that the rule is applied to $[x : \exists r.C, t_{x:\exists r.C}]$. Let $l := \mathsf{level}(x)$ and κ_l, κ'_l be the tuples associated with level l in $(\mathcal{A}, \mathcal{S})$ and $(\mathcal{A}', \mathcal{S}')$. Let y be the new variable in $\mathcal{A}'$ introduced by the rule. Obviously, it is $\mathsf{level}(y) = \mathsf{level}(x) + 1$.

   The first component of κ'_l remains unchanged. The second component decreases, because the existential restriction rule is not applicable to $[x : \exists r.C, t_{x:\exists r.C}]$ in $(\mathcal{A}', \mathcal{S}')$ any more. Therefore, an increase in the third component of κ'_l and in κ'_m for $m > l$ need not be considered. Since all components of κ'_m with $m < l$ remain unchanged, it is $\kappa(\mathcal{A}, \mathcal{S}) \succ \kappa(\mathcal{A}', \mathcal{S}')$.

5. **Supremum Restriction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\exists_2} (\mathcal{A}', \mathcal{S}')$
   Assume that the rule is applied to the assertions $[x : \exists r.C, t_{x:\exists r.C}]$, $[xy : r, t_{xy:r}]$. Let $l := \mathsf{level}(x)$ and κ_l, κ'_l be the tuples associated with level l in $(\mathcal{A}, \mathcal{S})$ and $(\mathcal{A}', \mathcal{S}')$.

   The first two components in κ'_l are not changed and the third component decreases, because the pair $[x : \exists r.C, t_{x:\exists r.C}]$, $[xy : r, t_{xy:r}]$ is no longer counted. Therefore, an increase in tuples κ'_m with $m > l$ (especially in κ'_{l+1}) is irrelevant. Since all components of κ'_m with $m < l$ remain unchanged, it is $\kappa(\mathcal{A}, \mathcal{S}) \succ \kappa(\mathcal{A}', \mathcal{S}')$.

6. **Value Restriction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\forall_1} (\mathcal{A}', \mathcal{S}')$
   Analogous to the existential restriction rule.

7. **Infimum Restriction:** $(\mathcal{A}, \mathcal{S}) \longrightarrow_{\forall_2} (\mathcal{A}', \mathcal{S}')$
   Analogous to the supremum restriction rule.

■

Lemma 14 *If some $\mathcal{E} = (\mathcal{A}, \mathcal{S})$ is complete, the applied optimization method determines the minimal possible value $\tau(t_{x_0:D})$ that is a solution for $\mathcal{S}$ in $\mathcal{E}$.*

Proof: The completion rules generate systems of inequations based on values in [0..1]. After some further treatment of the appearing $m_i^{-1}, \mathsf{min}, \mathsf{max}$ operators,

methods from the domain of linear programming, e.g. the *simplex method* (see [9]), are able to solve those inequations and to determine the smallest value that is a solution for $\tau(t_{x_0:D})$. There are different methods to remove min/max-equations. The simplest one is to split $\mathcal{S}$ into two sets $\mathcal{S}', \mathcal{S}''$ for each occurring min or max. In the case of min, the equation $t = \mathsf{min}(t_1, t_2)$ is substituted by $\{t = t_1, t \geq t_2\}$ in $\mathcal{S}'$ and $\{t = t_2, t \geq t_1\}$ in $\mathcal{S}''$. max is treated in the same manner. A more elegant way is to directly express one operator by two equations and only to split $\mathcal{S}$ into two sets for the remaining one. The single m_i^{-1}, divided into $m_{i_1}^{-1}, m_{i_2}^{-1}$, are treated by handing over their definition to the optimization method. Since only linear functions appear, their inverse function equation is easy to compute. ∎

Theorem 8 can now be proved by using the three lemmata.

Proof of Theorem 8:
Due to Lemma (13.4), the algorithm terminates. If C is not satisfiable to degree 1 (e.g. in the case $C = P \sqcap \neg P$), then there exists no solution of $\mathcal{S}_0$, and hence the output is 1 according to Definition 4.

Otherwise, let $t \in [0..1]$ be the output. We have to show that t is the degree t_D to which D is entailed by C.

By Lemma (13.2), there exists a model $(\mathcal{I}', \tau')$ with $\tau'(t_{x_0:D}) = t$, i.e., $t_D \leq t$.

Assume that there exists a model $(\mathcal{I}', \tau')$ of $\mathcal{E}_0$ with $\tau'(t_{x_0:D}) < t$. By Lemma (13.1), we obtain a complete extended ABox $\mathcal{E}$ such that there exists an expansion τ'' of τ', i.e., $\tau''(t_{x_0:D}) = \tau'(t_{x_0:D})$. Then, the optimization method yields a solution τ^* with $\tau^*(t_{x_0:D}) \leq \tau'(t_{x_0:D}) < t$. Thus, we obtain a contradiction because we assumed t to be the output of Algorithm 10. We get $t_D \geq t$ and hence $t_D = t$. ∎

5 Summary

This article presented the first steps into reasoning with vague information, or more precisely with concepts that are not characterized by crisp boundaries. The major feature is the incorporation of membership manipulators that catch the semantics of adjectives which have a major impact on some disciplines. There are different sensible applications imaginable that are based on the presented reasoning method. A concrete application from the medical domain combines medical terminology processing within a tutoring system for teaching findings in radiology. In this context, the main problem addresses the comparison between vague concept descriptions, i.e. the similarity between findings from students and those from experts. In a preprocessing step, the single statements from the findings are translated into concept definitions. The number and length of defined concepts is small but very strongly intermixed with manipulators. Therefore, $\mathcal{ALC}_{F_M}$ is a good tool for comparing them by capturing the inherent semantics of the underlying medical problem. Details of this work – especially about the used natural language preprocessing method – can be found in [11].

Open at this time remains the question of how to incorporate concrete fuzzy sets. For example, in the application above, it is sensible to compare statements like *enlarged* with *approximately 14 cm*, where both are defined by fuzzy sets. Thus, atomic concepts must be attachable to these fuzzy sets.

There are many other works in the area of reasoning with vagueness in medicine like [4, 5]. They only deal with concrete cases and data extensions by applying most often some heuristics. In contrast, the method presented in this paper is based on a logical foundation that guarantees sound and complete intensional inferences.

Extending this work, other inference tasks for $\mathcal{ALC}_{F_M}$, e.g. consistency of arbitrary $\mathcal{ALC}_{F_M}$-ABoxes, will be investigated. The proposed method is sound and complete but not really equipped to work on very large knowledge bases. Consequently, the question of complexity has to be studied in further work.

References

[1] F. Baader and U. Sattler. Number restrictions on complex roles in description logics. In *Proceedings of the Fifth International Conference on the Principles of Knowledge Representation and Reasoning (KR'96)*. Morgan Kaufmann, 1996.

[2] R. J. Brachman and J. G. Schmolze. An overview of the KL-ONE knowledge representation system. In *Cognitive Science, 9(2)*, 1985.

[3] D. Dubois, J. Lang, and H. Prade. Possibilistic Logic. In *Handbook of Logic in Artificial Intelligence and Logic Programming Vol. 3: Nonmonotonic Reasoning and Uncertain Reasoning*. Clarendon Press Oxford, 1994.

[4] A.O. Esogbue and R.C. Elder. Measurement and valuation of a fuzzy mathematical model for medical diagnosis. In *Fuzzy Sets and Systems 10*, 1983.

[5] M. Fathi and D. Meyer. MEDUSA - a fuzzy expert system for medical diagnosis of acute abdominal pain. In *Methods of Information in Medicine, No. 38*. Schattauer, 1994.

[6] J. Heinsohn. $\mathcal{ACLP}$: Ein hybrider Ansatz zur Modellierung von Unsicherheit in terminologischen Logiken. Infix, Dissertationen zur knstlichen Intelligenz, 1994.

[7] J. Heinson and B. Owsnicki-Klewe. Probabilistic inheritance and reasoning in hybrid knowledge representation systems. In *Proceedings of the 12th German Workshop on Artificial Intelligence (GWAI'88)*, 1988.

[8] M. Jaeger. Probabilistic reasoning in terminological logics. In *Principles of Knowledge Representation and Reasoning (KR'94)*. Morgan Kaufmann, 1994.

[9] G. V. Reklaitis, A. Ravindran, and K. M. Ragsdell. Engineering Optimization - Methods and Applications, chapter 4. Wiley, 1983.

[10] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. In *Journal of Artificial Intelligence, 48(1)*, 1991.

[11] C. B. Tresp and U. Tüben. Medical terminology processing for a tutoring system. In *International Conference on Computational Intelligence and Multimedia Applications (ICCIMA'98)*. 1998.

[12] L.A. Zadeh. A fuzzy-set-theoretic interpretation of linguistic hedges. In *Journal of Cybernetics, 2*, 1972.