A Formal Framework for Description Logics with Uncertainty ⋆

Volker Haarslev, Hsueh-Ieng Pai, Nematollaah Shiri *

Department of Computer Science and Software Engineering
Concordia University, Montreal, Quebec, Canada

Abstract

Description Logics (DLs) play an important role in the Semantic Web as the foundation of ontology language OWL DL. On the other hand, uncertainty is a form of deficiency or imperfection commonly found in real-world information/data. In this paper, we present a framework for knowledge bases with uncertainty expressed in the Description Logic $\mathcal{ALC}_U$, which is a propositionally complete representation language providing conjunction, disjunction, existential and universal quantifications, and full negation. The proposed framework is equipped with a constraint-based reasoning procedure that derives a collection of assertions as well as a set of linear/nonlinear constraints that encode the semantics of the uncertainty knowledge base. The interesting feature of our approach is that, by simply tuning the combination functions that generate the constraints, different notions of uncertainty can be modeled and reasoned with, using a single reasoning procedure. We establish soundness, completeness, and termination of the reasoning procedure. Detailed explanations and examples are included to describe the proposed completion rules.

Key words: Description Logics, Uncertainty, Knowledge Base, Tableau Procedure, Constraint Solving
PACS: 02.10.Ab, 02.50.Tt, 07.05.Mh, 89.20.Hh

⋆ This work is supported in part by Natural Sciences and Engineering Research Council (NSERC) of Canada, and by Concordia University.
* Corresponding Author. Address: Computer Science & Software Engineering, Concordia University. 1455 Maisonneuve West, Montreal, QC, H3G 1M8, Canada.
Email address: `shiri@cse.concordia.ca` (Nematollaah Shiri).

1 Introduction

The vision of the *Semantic Web* [5] was first introduced by Tim Berners-Lee as "a Web of data that can be processed directly or indirectly by machines" [4]. The idea is to make Web resources more machine-interpretable by giving them a well-defined meaning through semantic markups. One way to encode such semantic mark-ups is using ontologies. An ontology is "an explicit specification of a conceptualization" [12]. Informally, an ontology consists of a set of terms in a domain, the relationship between the terms, and a set of constraints imposed on the way in which those terms can be combined. Constraints such as concept conjunction, disjunction, negation, existential quantifier, and universal quantifier can all be expressed using ontology languages. By explicitly defining the relationships and constraints among the terms, the semantics of the terms can be better defined and understood.

Among the Semantic Web ontology languages, the OWL Web Ontology Language [40] is the most recent W3C Recommendation. One of its species, OWL DL, is named because of its correspondence with *Description Logics* (DLs) [1]. The family of DLs is mostly a subset of first-order logic (FOL) that is considered to be attractive because it keeps a good compromise between the expressive power and the computational tractability [1]. The well-defined semantics as well as the availability of the powerful reasoning tools make the family of DLs particularly interesting to the Semantic Web community [2].

The standard DLs, such as the one that is the basis of OWL DL, focus on the classical logic, which is more suitable to describe concepts that are crisp and well-defined in nature. However, in the real-world applications, *uncertainty*, which refers to a form of deficiency or imperfection in the information for which the truth of such information is not established definitely [23], is everywhere. Not only because the real-world information is mostly imperfect or deficient, but also because many realistic applications need the capability to handle uncertainty – from classification of genes in bioinformatics, schema matching in information integration, to matchmaking in Web services. The need to model and reason with uncertainty has been found in many different Semantic Web contexts. For example, in an online medical diagnosis system, one might want to find out to what degree a person, John, would have heart disease if the certainty that an obese person would have heart disease lies between 0.7 and 1, and John is obese with a degree between 0.8 and 1. Such knowledge cannot be expressed nor be reasoned with the standard DLs.

In this paper, we propose a decidable constraint-based resolution approach to reason with uncertainty expressed in the DL $\mathcal{ALC}_U$. This language extends the standard DL $\mathcal{ALC}$ [30] with uncertainty, and is propositionally complete with conjunction, disjunction, existential and universal quantifications, and full negation. Constraint-based reasoning [10] solves reasoning problems by stating constraints about the problem and then finding solution satisfying all the constraints. There are several advantages in our constraint-based approach. For instance, constraints have well-defined and often intuitive semantics making them suitable to express complex uncertainty constraints. Also, constraints are declarative and hence easy to generate and use in other modules. Besides, there are many constraint solvers and algorithms to process them [6].

The constraint-based reasoning procedure proposed in this paper derives a set of assertions and constraints that encode the semantics of the $\mathcal{ALC}_U$ knowledge base. These derived constraints are then solved using the constraint solver to perform the reasoning tasks. The interesting feature of this approach is that, by simply tuning the combination functions that generate the constraints, different notions of uncertainty can be modeled and reasoned with, using a single reasoning procedure.

This paper is an extension of our previous work as follows. In [13], we presented a basic framework for representing the uncertainty knowledge as well as an initial attempt to study the inference rules. In [14], we presented a reasoning procedure for dealing with acyclic uncertainty knowledge bases. In this paper, we further extend [14] by presenting a reasoning procedure for dealing with general (both cyclic and acyclic) uncertainty knowledge bases. In addition, we establish soundness, completeness, and termination of the proposed reasoning procedure.

The rest of this paper is organized as follows. Section 2 gives an overview of the DL $\mathcal{ALC}$ and other related work. Section 3 presents the DL $\mathcal{ALC}_U$, the proposed constraint-based tableau reasoning procedure, along with an illustrative example. We also establish the soundness, completeness, and termination of the $\mathcal{ALC}_U$ reasoning procedure. Finally, concluding remarks and future directions are presented in Section 4.

2 Background and Related Work

In this section, we first give an overview of the DL $\mathcal{ALC}$, which is the basis of the DL $\mathcal{ALC}_U$. We then review the related work.

2.1 Overview of the DL $\mathcal{ALC}$

Description logics (DLs) are a family of knowledge representation languages that can be used to represent the knowledge of an application domain using concept *descriptions* and have *logic*-based semantics [1,3]. The DL fragment that we focus in this paper is called $\mathcal{ALC}$, which corresponds to the propositional multi-modal logic $\mathbf{K_{(m)}}$ [29].

The $\mathcal{ALC}$ framework consists of three main components – the description language, the knowledge base, and the reasoning procedure.

(1) *$\mathcal{ALC}$ Description Language:* Every description language has elementary descriptions which include atomic concepts (unary predicates) and roles (binary predicates). Complex descriptions can then be built inductively from concept constructors. The description language $\mathcal{ALC}$ consists of a set of language constructors that are of practical interest. Specifically, let R be a role name, the syntax of a concept description (denoted C or D) in $\mathcal{ALC}$ is described as follows, where the name of each rule is given in parenthesis.

$C, D \rightarrow A$ (Atomic Concept) |
$\neg C$ (Concept Negation) |
$C \sqcap D$ (Concept Conjunction) |
$C \sqcup D$ (Concept Disjunction) |
$\exists R.C$ (Role Exists Restriction) |
$\forall R.C$ (Role Value Restriction)

For example, let *Person* be an atomic concept and *hasParent* be a role. Then $\forall hasParent.Person$ is a concept description. We use Top Concept $\top$ as a synonym for $A \sqcup \neg A$, and Bottom Concept $\perp$ as a synonym for $A \sqcap \neg A$.

The semantics of the description language is defined using the notion of interpretation. An interpretation $\mathcal{I}$ is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty domain of the interpretation, and $\cdot^{\mathcal{I}}$ is an interpretation function that maps each atomic concept A to a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, each atomic role R to a binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and each individual name a to an element individual name a to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. The interpretations of concept descriptions are shown below:

$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$
$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$(\exists R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} | \; \exists b \in \Delta^{\mathcal{I}} : (a, b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}$
$(\forall R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} | \; \forall b \in \Delta^{\mathcal{I}} : (a, b) \in R^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\}$

(2) *$\mathcal{ALC}$ Knowledge Base:* The knowledge base is composed of a Terminological Box (TBox) and an Assertional Box (ABox). A TBox $\mathcal{T}$ is a set of statements about how concepts in an application domain are related to

each other. Let C and D be concept descriptions. The TBox is a finite, possibly empty, set of terminological axioms that could be a combination of *concept inclusions* of the form $\langle C \sqsubseteq D \rangle$ (that is, C is subsumed by D) and *concept equations* of the form $\langle C \equiv D \rangle$ (that is, C is equivalent to D). For example, the axiom $\langle ObesePerson \equiv Person \sqcap Obese \rangle$ states that the concept ObesePerson is equivalent to the conjunction of concepts Person and Obese. An interpretation $\mathcal{I}$ satisfies $\langle C \sqsubseteq D \rangle$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, and it satisfies $\langle C \equiv D \rangle$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$. An interpretation $\mathcal{I}$ satisfies a TBox $\mathcal{T}$ iff $\mathcal{I}$ satisfies every axiom in $\mathcal{T}$.

An ABox is a set of statements that describe a specific state of affairs in an application domain, with respect to some individuals, in terms of concepts and roles. Let a and b be individuals, C be a concept, R be a role, and let ":" denote "is an instance of". An ABox includes of a set of assertions that could be a combination of concept assertions of the form $\langle a : C \rangle$ and role assertions of the form $\langle (a, b) : R \rangle$. For example, the concept assertion $\langle \mathit{John} : \mathit{ObesePerson} \rangle$ asserts that individual *John* is an instance of concept *ObesePerson*. Similarly, the role assertion $\langle (\mathit{John}, \mathit{Mary}) : \mathit{hasMother} \rangle$ asserts that *John*'s mother is *Mary*. An interpretation $\mathcal{I}$ satisfies $\langle a : C \rangle$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, and it satisfies $\langle (a, b) : R \rangle$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$. An interpretation $\mathcal{I}$ satisfies an ABox $\mathcal{A}$, iff it satisfies every assertion in $\mathcal{A}$ with respect to a TBox $\mathcal{T}$.

An interpretation $\mathcal{I}$ *satisfies* (or is a *model* of) a knowledge base $\Sigma = \langle \mathcal{T}, \mathcal{A} \rangle$ (denoted $\mathcal{I} \models \Sigma$), iff it satisfies both components of Σ. The knowledge base Σ is *consistent* if there exists an interpretation $\mathcal{I}$ that satisfies Σ. We say that Σ is *inconsistent* otherwise.

(3) *$\mathcal{ALC}$ Reasoning Procedure:* Most DL systems use tableau-based reasoning procedure (called tableau algorithm) to provide reasoning services [1]. The main reasoning services include (i) the consistency problem which checks if the ABox is consistent with respect to the TBox, (ii) the entailment problem which checks if an assertion is entailed by a knowledge base, (iii) the concept satisfiability problem which checks if a concept is satisfiable with respect to a TBox, and (iv) the subsumption problem which checks if a concept is subsumed by another concept with respect to a TBox. All these reasoning services can be reduced to the consistency problem [1]. The tableau algorithm can be used to check consistency of the knowledge base Σ. It tries to construct a model by iteratively applying a set of so-called completion rules in arbitrary order. Each completion rule application adds one or more additional inferred assertions to the ABox to make it explicit the knowledge that was previously present implicitly. The algorithm terminates when no further completion rule is applicable. If one could arrive a completion that contains no contradiction (also known as clash), then the knowledge base is consistent. Otherwise, the knowledge base is inconsistent.

2.2 Related Work

Incorporating uncertainty in DL frameworks has been the topic of numerous research for more than a decade [7,9,11,17,18,20,21,24,25,27,31,32,34–39]. Based on the underlying mathematical foundation and the type of uncertainty modeled, we can classify each proposal into one of the three approaches: fuzzy, probabilistic, and possibilistic approach.

The fuzzy approach [7,17,31–39], based on fuzzy set theory [41], deals with the vagueness in the knowledge, where a proposition is true only to some degree. For example, the statement "Jason is obese with degree 0.4" indicates Jason is slightly obese. Here, the value 0.4 is the degree of membership that Jason is in concept obese.

The probabilistic approach [9,11,20,21,24,25], based on the classical probability theory, deals with the uncertainty due to lack of knowledge, where a proposition is either true or false, but one does not know for sure which one is the case. Hence, the certainty value refers to the probability that the proposition is true. For example, one could state that: "The probability that Jason would have heart disease given that he is obese lies in the range $[0.8, 1]$."

Finally, the possibilistic approach [18,27], based on possibility theory [42], allows both certainty (necessity measure) and possibility (possibility measure) be handled in the same formalism. For example, by knowing that "Jason's weight is above 80 kg", the proposition "Jason's weight is at least 80 kg" is necessarily true with certainty 1, while "Jason's weight is 90 kg" is possibly true with certainty 0.5.

What sets our approach apart from the existing approaches is the way knowledge bases are reasoned. There have been a number of approaches proposed on supporting uncertainty/DL reasoning. Some extended the tableau-based reasoning procedure used in standard DLs, some transformed the uncertainty knowledge bases into standard DL knowledge bases, while others employed completely different reasoning procedures such as the inference algorithm developed for Bayesian networks. A survey of these frameworks can be found in Chapter 6 of [1] and in [16].

Although constraint-based reasoning procedures were proposed in [7,36,38,39], there are some major differences between these works and the one we present in this paper. While our approach is to develop one reasoning procedure for dealing with uncertainty with different mathematical foundations, others mainly considered one form. For instance, [39] supports only fuzzy logic with Zadeh semantics, [7] supports only product t-norm, and [38] supports only Lukasiewicz semantics. Although [36] supports both Zadeh and Lukasiewicz semantics, it uses two sets of reasoning procedures instead of using one generic reasoning procedure to deal with different semantics. Another difference is that the reasoning procedure we present in this paper supports general TBoxes (i.e., concept descriptions are allowed to appear on the left hand side of an axiom, and cyclic axioms are supported), which is more complicated than the ones considered in [36,39].

3 The $\mathcal{ALC}_U$ Framework

In this section, we present the $\mathcal{ALC}_U$ framework, which extends the standard $\mathcal{ALC}$ framework with uncertainty. To support uncertainty, each component of the standard $\mathcal{ALC}$ framework needs to be extended. For this, we first introduce the DL $\mathcal{ALC}_U$, including the syntax and semantics of the description language and the knowledge base. We then present the reasoning procedure and establish its correctness. After that, we illustrate through examples the various extended components of the $\mathcal{ALC}_U$ framework.

3.1 The Description Language $\mathcal{ALC}_U$

Recall that the description language refers to the language used for building concepts. The syntax of the $\mathcal{ALC}_U$ description language is identical to that of the standard $\mathcal{ALC}$, while the corresponding semantics is extended with uncertainty.

We assume that the certainty values form a complete lattice $\mathcal{L} = \langle \mathcal{V}, \preceq \rangle$, where $\mathcal{V}$ is the certainty domain, and $\preceq$ is the partial order on $\mathcal{V}$. Also, $\prec$, $\succeq$, $\succ$, and $=$ are used with their obvious meanings. We use l to denote the least element in $\mathcal{V}$, t for the greatest element in $\mathcal{V}$, $\oplus$ for the join operator (the least upper bound) in $\mathcal{L}$, $\otimes$ for the meet operator (the greatest lower bound), and $\sim$ for the negation operator. We also assume that there is only one underlying certainty lattice for the entire knowledge base. An advantage of using a lattice is that it can be used to model both qualitative and quantitative certainty values. An example for the former is the classical logic with lattice $\mathcal{L} = \langle \{0, 1\}, \leq \rangle$, where $\leq$ is the usual order on binary values $\{0, 1\}$. For the latter, an example would be a family of multi-valued logics over the unit interval [0, 1], such as fuzzy logic, with certainty lattice $\mathcal{L} = \langle [0, 1], \leq \rangle$.

The semantics of the description language is based on the notion of an interpretation. An interpretation $\mathcal{I}$ is defined as a pair $(\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is the domain and $\cdot^\mathcal{I}$ is an interpretation function that maps each

- atomic concept A into a certainty function CF_A, where $CF_A : \Delta^\mathcal{I} \rightarrow \mathcal{V}$

- atomic role R into a certainty function CF_R, where $CF_R : \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \rightarrow \mathcal{V}$
- individual name a to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$

where $\mathcal{V}$ is the certainty domain. For example, let *John* be an individual name and *Obese* be an atomic concept. Then, $Obese^{\mathcal{I}}(John^{\mathcal{I}})$ gives the certainty that *John* is an instance of the concept *Obese*. The syntax and semantics of the description language $\mathcal{ALC}_U$ are summarized in Table 1.

Table 1
Syntax and Semantics of the Description Language $\mathcal{ALC}_U$

Name	**Syntax**	**Semantics** ($a \in \Delta^{\mathcal{I}}$)
Top Concept	$\top$	$\top^{\mathcal{I}}(a) = t$
Bottom Concept	$\bot$	$\bot^{\mathcal{I}}(a) = l$
Concept Negation	$\neg C$	$(\neg C)^{\mathcal{I}}(a) = \sim C^{\mathcal{I}}(a)$
Concept Conjunction	$C \sqcap D$	$(C \sqcap D)^{\mathcal{I}}(a) = f_c(C^{\mathcal{I}}(a), D^{\mathcal{I}}(a))$
Concept Disjunction	$C \sqcup D$	$(C \sqcup D)^{\mathcal{I}}(a) = f_d(C^{\mathcal{I}}(a), D^{\mathcal{I}}(a))$
Role Exists Restriction	$\exists R.C$	$(\exists R.C)^{\mathcal{I}}(a) = \oplus_{b \in \Delta^{\mathcal{I}}}\{f_c(R^{\mathcal{I}}(a,b), C^{\mathcal{I}}(b))\}$
Role Value Restriction	$\forall R.C$	$(\forall R.C)^{\mathcal{I}}(a) = \otimes_{b \in \Delta^{\mathcal{I}}}\{f_d(\sim R^{\mathcal{I}}(a,b), C^{\mathcal{I}}(b))\}$

As shown in Table 1, the certainty of the Top Concept $\top$ is the greatest element in the certainty lattice, t. Similarly, the certainty of the Bottom Concept $\bot$ is the least element in the certainty lattice, l.

The operator $\sim$ in Table 1 denotes the negation function, where $\sim : \mathcal{V} \rightarrow \mathcal{V}$ must satisfy the following properties:

- Boundary Conditions: $\sim l = t$ and $\sim t = l$.
- Double Negation: $\sim(\sim \alpha) = \alpha$, for all $\alpha \in \mathcal{V}$.

The negation operator $\sim$ in the certainty lattice is used as the default negation function. That is, $(\neg C)^{\mathcal{I}}(a) = \sim C^{\mathcal{I}}(a)$, for all $a \in \Delta^{\mathcal{I}}$. A common interpretation of $\neg C$ is $1 - C^{\mathcal{I}}(a)$. For example, if the certainty domain is $\mathcal{V} = [0, 1]$, and if the certainty that individual John is Obese is 0.8. Then, the certainty that John is not Obese is $1 - 0.8 = 0.2$.

In addition, f_c and f_d in Table 1 denote the conjunction and disjunction functions, respectively, both of which we refer as the *combination functions*. They are used to specify how one should interpret a given description language. A combination function f is a binary function from $\mathcal{V} \times \mathcal{V}$ to $\mathcal{V}$. This function combines a pair of certainty values into one. A combination function must satisfy some properties as listed in Table 2 [22].

A *conjunction function* f_c is a combination function that satisfies properties P_1, P_2, P_5, P_6, P_7, and P_8 as described in Table 2. The monotonicity

Table 2
Combination Function Properties

ID	Property Name	Property Definition
P_1	Monotonicity	$f(\alpha_1, \alpha_2) \preceq f(\beta_1, \beta_2)$ if $\alpha_i \preceq \beta_i$, for $i = 1, 2$
P_2	Bounded Above	$f(\alpha_1, \alpha_2) \preceq \alpha_i$, for $i = 1, 2$
P_3	Bounded Below	$f(\alpha_1, \alpha_2) \succeq \alpha_i$, for $i = 1, 2$
P_4	Boundary Condition (Above)	$\forall \alpha \in \mathcal{V},\ f(\alpha, l) = \alpha$ and $f(\alpha, t) = t$
P_5	Boundary Condition (Below)	$\forall \alpha \in \mathcal{V},\ f(\alpha, t) = \alpha$ and $f(\alpha, l) = l$
P_6	Continuity	f is continuous w.r.t. each of its arguments
P_7	Commutativity	$\forall \alpha, \beta \in \mathcal{V},\ f(\alpha, \beta) = f(\beta, \alpha)$
P_8	Associativity	$\forall \alpha, \beta, \delta \in \mathcal{V},\ f(\alpha, f(\beta, \delta)) = f(f(\alpha, \beta), \delta)$

property asserts that increasing the certainties of the arguments in f improves the certainty that f returns. The bounded value and boundary condition properties are included so that the interpretation of the certainty values makes sense. The commutativity property allows reordering of the arguments of f, say for optimization purposes. Finally, the associativity of f ensures that different evaluation orders of concept conjunctions will not yield different results. Some common conjunction functions are the well-known minimum function, the algebraic product ($prod(x, y) = x \cdot y$) and the bounded difference ($bDiff(x, y) = max(0, x + y - 1)$).

A *disjunction function* f_d is a combination function that satisfies properties P_1, P_3, P_4, P_6, P_7, and P_8 as described in Table 2. These properties are enforced for similar reasons as in the conjunction case. Some common disjunction functions are the maximum function, the probability independent function ($ind(x, y) = x + y - x \cdot y$) and the bounded sum function ($bSum(x, y) = min(1, x + y)$).

In Table 1, the semantics of the Role Exists Restriction $\exists R.C$ is defined as $(\exists R.C)^{\mathcal{I}}(a) = \oplus_{b \in \Delta^{\mathcal{I}}}\{f_c(R^{\mathcal{I}}(a, b), C^{\mathcal{I}}(b))\}$, for all $a \in \Delta^{\mathcal{I}}$. The intuition here is that $\exists R.C$ is viewed as the open first order formula $\exists b.\ R(a, b) \wedge C(b)$, where $\exists$ is viewed as a disjunction over certainty values associated with $R(a, b) \wedge C(b)$. Specifically, the semantics of $R(a, b) \wedge C(b)$ is captured using the conjunction function $f_c(R^{\mathcal{I}}(a, b), C^{\mathcal{I}}(b))$, and $\exists b$ is captured using the join operator in the certainty lattice $\oplus_{b \in \Delta^{\mathcal{I}}}$.

Similarly, the semantics of the Role Value Restriction $\forall R.C$ is defined as $(\forall R.C)^{\mathcal{I}}(a) = \otimes_{b \in \Delta^{\mathcal{I}}}\{f_d(\sim R^{\mathcal{I}}(a, b), C^{\mathcal{I}}(b))\}$, for all $a \in \Delta^{\mathcal{I}}$. The intuition is that $\forall R.C$ is viewed as the open first order formula $\forall b.\ R(a, b) \rightarrow C(b)$, where $R(a, b) \rightarrow C(b)$ is equivalent to $\neg R(a, b) \vee C(b)$, and $\forall$ is viewed as a conjunction over certainty values associated with the implication $R(a, b) \rightarrow C(b)$. To be more precise, the semantics of $R(a,b) \rightarrow C(b)$ is captured using the disjunction and the negation functions as $f_d(\sim R^{\mathcal{I}}(a,b), C^{\mathcal{I}}(b))$, and $\forall b$ is captured using the meet operator in the certainty lattice $\otimes_{b \in \Delta^{\mathcal{I}}}$.

We say a concept is in *negation normal form* (NNF) if the negation operator appears only in front of concept names. The following two inter-constructor properties allow the transformation of concept descriptions into NNFs.

- De Morgan's Rule: $\neg(C \sqcup D) \equiv \neg C \sqcap \neg D$ and $\neg(C \sqcap D) \equiv \neg C \sqcup \neg D$.
- Negating Quantifiers Rule: $\neg\exists R.C \equiv \forall R.\neg C$ and $\neg\forall R.C \equiv \exists R.\neg C$.

3.2 $\mathcal{ALC}_U$ Knowledge Base

The *knowledge base* Σ in the $\mathcal{ALC}_U$ framework is a pair $\langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox. An interpretation $\mathcal{I}$ *satisfies* (or is a *model* of) Σ (denoted $\mathcal{I} \models \Sigma$), if and only if it satisfies both $\mathcal{T}$ and $\mathcal{A}$. The knowledge base Σ is *consistent* if there exists an interpretation $\mathcal{I}$ that satisfies Σ, and is *inconsistent* otherwise.

3.2.1 $\mathcal{ALC}_U$ TBox

An $\mathcal{ALC}_U$ TBox $\mathcal{T}$ consists of a set of terminological axioms defining how concepts are related to each other. Each axiom is associated with a certainty value as well as a conjunction function and a disjunction function which are used to interpret the concept descriptions in the axiom. Specifically, an $\mathcal{ALC}_U$ TBox consists of axioms that could be a combination of concept inclusions of the form $\langle C \sqsubseteq D \mid \alpha, f_c, f_d \rangle$ and concept equations of the form $\langle C \equiv D \mid \alpha, f_c, f_d \rangle$, where C and D are concept descriptions, $\alpha \in \mathcal{V}$ is the certainty that the axiom holds, and f_c and f_d are the combination functions used to interpret the concepts that appear in the axiom. In particular, f_c is the conjunction function used as the semantics of concept conjunction and part of the role exists restriction, and f_d is the disjunction function used as the semantics of concept disjunction and part of the role value restriction. The concept equation $\langle C \equiv D \mid \alpha, f_c, f_d \rangle$ is equivalent to $\langle (C \sqsubseteq D) \sqcap (D \sqsubseteq C) \mid \alpha, f_c, f_d \rangle$.

For example, the axiom $\langle Rich \sqsubseteq ((\exists owns.ExpensiveCar \sqcup \exists owns.Airplane)$ $\sqcap Golfer) \mid [0.8, 1], min, max \rangle$ states that the concept *Rich* is subsumed by owning expensive car or owning an airplane, and being a golfer. The certainty of this axiom is at least 0.8, with all the concept conjunctions interpreted using *min* function, and all the concept disjunctions interpreted using *max*.

All axioms can be transformed into their normal forms, that is, axioms of the form $\langle \top \sqsubseteq \ldots \mid \alpha, f_c, f_d \rangle$. For example, a concept inclusion $\langle C \sqsubseteq D \mid \alpha, f_c, f_d \rangle$

has the normal form $\langle \top \sqsubseteq \neg C \sqcup D \mid \alpha, f_c, f_d \rangle$. For such transformation to make sense, the semantics of the concept inclusion is restricted to $f_d(\sim C^{\mathcal{I}}(a), D^{\mathcal{I}}(a))$, for all $a \in \Delta^{\mathcal{I}}$, where $\sim C^{\mathcal{I}}(a)$ captures the semantics of $\neg C$, and f_d captures the semantics of $\sqcup$ in $\neg C \sqcup D$. Hence, an interpretation $\mathcal{I}$ satisfies $\langle C \sqsubseteq D \mid \alpha, f_c, f_d \rangle$ if $f_d(\sim C^{\mathcal{I}}(a), D^{\mathcal{I}}(a)) = \alpha$, for all $a \in \Delta^{\mathcal{I}}$.

Note 1 *Currently, the description language constructors used in $\mathcal{ALC}_U$ TBoxes are kept the same as the standard $\mathcal{ALC}$ counterpart, and the only difference between an $\mathcal{ALC}_U$ axiom and an $\mathcal{ALC}$ axiom is that each axiom is extended with the uncertainty parameters (i.e., a certainty value and a pair of combination functions). However, existing probabilistic DL frameworks such as [11] divide the TBoxes into two parts: the standard axioms (which contains no probabilistic knowledge) and the conditional constraints. Since supporting conditional constraints in DL requires syntactical extension by introducing new language constructor, $(C \mid D)$, our framework currently does not support conditional constraints. Note that although one may set combination functions to simulate probabilistic reasoning, the interpretation of concept inclusion as material implication may yield unintuitive results.*

3.2.2 $\mathcal{ALC}_U$ ABox

An $\mathcal{ALC}_U$ ABox $\mathcal{A}$ consists of a set of assertions, each of which is associated with a certainty value and a pair of combination functions used to interpret the concept description(s) in the assertion. Specifically, these assertions could include concept assertions of the form $\langle a : C \mid \alpha, f_c, f_d \rangle$ and role assertions of the form $\langle (a, b) : R \mid \alpha, -, - \rangle$, where a and b are individuals, C is a concept, R is a role, $\alpha \in \mathcal{V}$, f_c is the conjunction function, f_d is the disjunction function, and $-$ denotes that the corresponding combination function is not applicable.

For instance, the assertion "Mary is tall and thin with degree between 0.6 and 0.8" can be expressed as $\langle Mary : Tall \sqcap Thin \mid [0.6, 0.8], min, - \rangle$. Here, the concept conjunction is interpreted using the *min* function, and the disjunction function is not applicable since there is no concept disjunction in this assertion. Hence, "$-$" is used as a place holder.

In terms of the semantics of the assertions, an interpretation $\mathcal{I}$ satisfies $\langle a : C \mid \alpha, f_c, f_d \rangle$ (resp. $\langle (a, b) : R \mid \alpha, -, - \rangle$) if $C^{\mathcal{I}}(a^{\mathcal{I}}) = \alpha$ (resp. $R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) = \alpha$).

There are two types of individuals that could be in an ABox - defined individuals and generated individuals, defined as follows. We also introduce the notion of predecessor and ancestor in Definition 2.

Definition 1 *(Defined/Generated Individual) Let* $\mathbf{I}$ *be the set of all individuals in an ABox. We call individuals whose names explicitly appear in the input ABox "defined individuals"* ($\mathbf{I_D}$)*, and those generated by the reasoning proce-*

dure "generated individuals" ($\mathbf{I_G}$)*. Note that* $\mathbf{I_D} \cap \mathbf{I_G} = \emptyset$*, and* $\mathbf{I_D} \cup \mathbf{I_G} = \mathbf{I}$*.*

Definition 2 *(Predecessor/Ancestor) An individual a is a "predecessor" of an individual b (or b is a R-successor of a) if the ABox $\mathcal{A}$ contains the assertion $\langle (a,b) : R \mid \alpha, -, - \rangle$. An individual a is an "ancestor" of b if it is either a predecessor of b or there exists a chain of assertions $\langle (a,b_1) : R_1 \mid \alpha_1, -, - \rangle$, $\langle (b_1,b_2) : R_2 \mid \alpha_2, -, - \rangle$,..., $\langle (b_k,b) : R_{k+1} \mid \alpha_{k+1}, -, - \rangle$ in $\mathcal{A}$.*

3.3 $\mathcal{ALC}_U$ Reasoning Procedure

Let $\Sigma = \langle \mathcal{T}, \mathcal{A} \rangle$ be an $\mathcal{ALC}_U$ knowledge base. Fig. 1 gives an overview of our constraint-based tableau reasoning procedure for $\mathcal{ALC}_U$. The rectangles represent data or knowledge bases, the arrows show the data flow, and the gray rounded boxes show where data processing is performed.

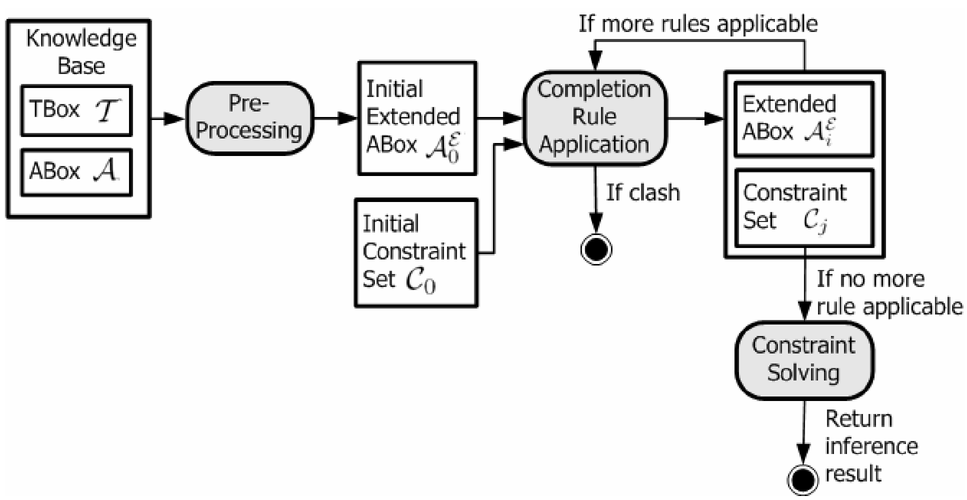

Fig. 1. Reasoning Procedure for $\mathcal{ALC}_U$

In what follows, we present the $\mathcal{ALC}_U$ tableau algorithm in detail. We first introduce the reasoning services offered, and then present the pre-processing phase and the completion rules. We also establish correctness of our $\mathcal{ALC}_U$ tableau algorithm.

3.3.1 $\mathcal{ALC}_U$ Reasoning Services

The $\mathcal{ALC}_U$ reasoning services include the consistency, the entailment, and the subsumption problems as described below.

Consistency Problem: To check if an $\mathcal{ALC}_U$ knowledge base $\Sigma = \langle \mathcal{T}, \mathcal{A} \rangle$ is consistent, we first apply the pre-processing steps (see Section 3.3.2) to obtain the initial extended ABox, $\mathcal{A}_0^{\mathcal{E}}$. In addition, the constraints set $\mathcal{C}_0$ is initialized to the empty set $\{\}$. We then apply the completion rules (see Section 3.3.3) to derive implicit knowledge from explicit ones. Through the application of each rule, we add any assertions that are derived to the extended ABox $\mathcal{A}_i^{\mathcal{E}}$. In addition, constraints which denote the semantics of the assertions are added to the constraints set $\mathcal{C}_j$, in the form of linear or nonlinear inequations. The completion rules are applied in arbitrary order as long possible, until either $\mathcal{A}_i^{\mathcal{E}}$ contains a clash or no further rule could be applied to $\mathcal{A}_i^{\mathcal{E}}$. If $\mathcal{A}_i^{\mathcal{E}}$ contains a clash, the knowledge base is inconsistent. Otherwise, the system of inequations in $\mathcal{C}_j$ is fed into the constraint solver to check its solvability. If the system of inequations is unsolvable, the knowledge base is inconsistent. Otherwise, the knowledge base is consistent.

Entailment Problem: Given an $\mathcal{ALC}_U$ knowledge base Σ, the entailment problem determines the degree to which an assertion X is true. Like in standard DLs, the entailment problem can be reduced to the consistency problem. That is, let X be an assertion of the form $\langle a : C \mid x_{a:C}, f_c, f_d\rangle$. The degree that Σ entails X is the degree of $x_{a:C}$ such that $\Sigma \cup \{\langle a : \neg C,\ x_{a:\neg C}\rangle\langle f_c, f_d\rangle\}$ is consistent.

Subsumption Problem: Let $\Sigma = \langle \mathcal{T}, \mathcal{A}\rangle$ be an $\mathcal{ALC}_U$ knowledge base, and $\langle C \sqsubseteq D \mid x_{C\sqsubseteq D}, f_c, f_d\rangle$ be the subsumption relationship to be checked. The subsumption problem determines the degree to which C is subsumed by D with respect to the TBox $\mathcal{T}$. Like in standard DLs, this problem can be reduced to the consistency problem by finding the degree of $x_{a:\neg C\sqcup D}$ such that $\Sigma \cup \{\langle a : C \sqcap \neg D \mid x_{a:C\sqcap\neg D}, f_c, f_d\rangle\}$ is consistent, where a is a new, generated individual name.

As in standard DLs, the model being constructed by the $\mathcal{ALC}_U$ tableau algorithm can be thought of as a forest. In what follows, we define a few related terms.

Definition 3 *(Forest, Node Label, Node Constraint, Edge Label) A "forest" is a collection of trees, with nodes corresponding to individuals, edges corresponding to relationships/roles between individuals, and root nodes corresponding to individuals present in the initial extended ABox. Each node is associated with a "node label", $\mathcal{L}(individual)$, to show the concept assertions associated with a particular individual, as well as a "node constraint", $\mathcal{C}(individual)$, for the corresponding constraints. Unlike in the standard DL where each element in the node label is a concept, each element in our node label is a quadruple, $\langle Concept, Certainty, f_c, f_d\rangle$. Finally, unlike in the standard DL where each edge is labeled with a role name, each edge in our case is associated with an "edge label", $\mathcal{L}(\langle individual_1, individual_2\rangle)$ which consists of a pair of elements $\langle Role, Certainty\rangle$. In case the certainty is a variable, "$-$" is used as a place holder.*

To present the $\mathcal{ALC}_U$ tableau algorithm in detail, we need to introduce a few concepts as follows.

Note 2 *In standard DLs, a TBox is unfoldable if one could eliminate all the defined names from the right hand side of all the axioms by substituting all the concept names with their equivalent definitions [1]. For example, consider the axioms $\langle A \equiv B \sqcap \exists R.C \rangle$ and $\langle D \equiv A \sqcup E \rangle$. Through the process of unfolding, we can replace the definition of D by $\langle D \equiv (B \sqcap \exists R.C) \sqcup E \rangle$. However, the idea of unfolding no longer works when uncertainty is present, since each axiom is associated with a certainty value and a pair of combination functions. For example, consider these two axioms in $\mathcal{ALC}_U$ which extended the $\mathcal{ALC}$ axioms with certainty values and combination functions: $\langle A \equiv B \sqcap \exists R.C \mid 0.6, min, max \rangle$ and $\langle D \equiv A \sqcup E \mid 0.7, \times, ind \rangle$. We can not simply replace A on the right hand side of the concept definition D with the definition of A, since there is a certainty value (0.6) and two combination functions $\langle min, max \rangle$ associated with the concept definition A. This example shows that unfolding may not be applicable in $\mathcal{ALC}_U$.*

Definition 4 *(Evaluation) Let $Var(\mathcal{C})$ be the set of certainty variables occurring in the constraints set $\mathcal{C}$, and $\mathcal{V}$ be the certainty domain. If the system of inequations in $\mathcal{C}$ is solvable, the solution to the constraints set $\pi : Var(\mathcal{C}) \rightarrow \mathcal{V}$ is called an "evaluation".*

Definition 5 *(Complete) An extended ABox $\mathcal{A}_c^{\mathcal{E}}$ is complete if no more completion rule can be applied to $\mathcal{A}_c^{\mathcal{E}}$ and the set of constraints $\mathcal{C}$ obtained during the rule application is solvable.*

Definition 6 *(Model) Let $\Sigma = \langle \mathcal{T}, \mathcal{A} \rangle$ be an $\mathcal{ALC}_U$ knowledge base, and $\mathcal{A}_c^{\mathcal{E}}$ be the extended ABox obtained by applications of the completion rules to the extended ABox $\mathcal{A}_i^{\mathcal{E}}$. Also, let $\mathcal{I}$ be an interpretation, π be an evaluation, α be a certainty value in the certainty domain, and x_X be the variable representing the certainty of assertion X. The pair $\langle \mathcal{I}, \pi \rangle$ is a model of the extended ABox $\mathcal{A}_c^{\mathcal{E}}$ if all the following hold:*

- *for each assertion $\langle a : C \mid \alpha, f_c, f_d \rangle \in \mathcal{A}_c^{\mathcal{E}}$, $C^{\mathcal{I}}(a) = \alpha$.*
- *for each assertion $\langle a : C \mid x_{a:C}, f_c, f_d \rangle \in \mathcal{A}_c^{\mathcal{E}}$, $C^{\mathcal{I}}(a) = \pi(x_{a:C})$.*
- *for each assertion $\langle (a, b) : R \mid \alpha, -, - \rangle \in \mathcal{A}_c^{\mathcal{E}}$, $R^{\mathcal{I}}(a, b) = \alpha$.*
- *for each assertion $\langle (a, b) : R \mid x_{(a,b):R}, -, - \rangle \in \mathcal{A}_c^{\mathcal{E}}$, $R^{\mathcal{I}}(a, b) = \pi(x_{(a,b):R})$.*

The knowledge base Σ is consistent if there exists a model for the extended ABox $\mathcal{A}_c^{\mathcal{E}}$.

3.3.2 Pre-processing Phase

The $\mathcal{ALC}_U$ tableau algorithm starts by applying the following pre-processing steps, which maintains the equivalence of the result with the original knowledge base.

(1) Replace each axiom of the form $\langle C \equiv D \mid \alpha, f_c, f_d\rangle$ with $\langle (C \sqsubseteq D) \sqcap (D \sqsubseteq C) \mid \alpha, f_c, f_d\rangle$. Note that, like in standard DLs, $(C \sqsubseteq D) \sqcap (D \sqsubseteq C)$ is not considered to be cyclic since it is equivalent to $(C \equiv D)$.
(2) Transform every axiom in the TBox into its normal form. That is, axioms of the form $\langle \top \sqsubseteq \ldots \mid \alpha, f_c, f_d\rangle$.
(3) Transform every concept (the TBox and the ABox) into its NNF. Let C and D be concepts, and R be a role. The NNF can be obtained by applying the following rules:
   - $\neg\neg(C) \equiv C$
   - $\neg(C \sqcup D) \equiv \neg C \sqcap \neg D$
   - $\neg(C \sqcap D) \equiv \neg C \sqcup \neg D$
   - $\neg\exists R.C \equiv \forall R.\neg C$
   - $\neg\forall R.C \equiv \exists R.\neg C$
(4) Augment the ABox $\mathcal{A}$ with respect to the TBox $\mathcal{T}$. That is, for each individual a in $\mathcal{A}$ and each axiom of the form $\langle \top \sqsubseteq C \mid \alpha, f_c, f_d\rangle$ in $\mathcal{T}$, add $\langle a : C \mid \alpha, f_c, f_d\rangle$ to $\mathcal{A}$.

We call the resulting ABox after the pre-processing phase the *initial extended ABox*, denoted by $\mathcal{A}_0^{\mathcal{E}}$.

3.3.3 $\mathcal{ALC}_U$ Completion Rules

In the standard $\mathcal{ALC}$, if $\mathcal{T}$ is an unfoldable TBox, one can always reduce a reasoning problem with respect to $\mathcal{T}$ to a reasoning problem with respect to the empty TBox [1]. More specifically, the TBox could be discarded after the pre-processing phase. However, as explained earlier in Note 2, the idea of unfolding is not applicable for $\mathcal{ALC}_U$. Hence, we need to keep the TBox during the completion rule application phase, and make use of the TBox whenever a new individual is added to the extended ABox. This may lead to nontermination of completion-rule applications. To ensure termination, we introduce the notion of blocking.

Definition 7 *(Blocking) Let $a, b \in \mathbf{I_G}$ be generated individuals in the extended ABox $\mathcal{A}_i^{\mathcal{E}}$, $\mathcal{A}_i^{\mathcal{E}}(a)$ and $\mathcal{A}_i^{\mathcal{E}}(b)$ be all the concept assertions for a and b in $\mathcal{A}_i^{\mathcal{E}}$. An individual b is blocked by some ancestor a (or a is the blocking individual for b) if $\mathcal{A}_i^{\mathcal{E}}(b) \subseteq \mathcal{A}_i^{\mathcal{E}}(a)$.*

Let $\mathcal{T}$ be the TBox obtained after the pre-processing phase, $\mathcal{A}_0^{\mathcal{E}}$ be the initial extended ABox, and $\mathcal{C}_0$ be the initial constraints set. Also, let α and β be certainty values, and Γ be either a certainty value in the certainty domain or the variable x_X denoting the certainty of assertion X. The $\mathcal{ALC}_U$ completion rules are defined as follows.

Clash Triggers:

$\langle a : \bot \mid \alpha, -, - \rangle \in \mathcal{A}_i^{\mathcal{E}}$, with $\alpha \succ l$
$\langle a : \top \mid \alpha, -, - \rangle \in \mathcal{A}_i^{\mathcal{E}}$, with $\alpha \prec t$
$\{\langle a : A \mid \alpha, -, - \rangle, \langle a : A \mid \beta, -, - \rangle\} \subseteq \mathcal{A}_i^{\mathcal{E}}$, with $\otimes(\alpha, \beta) = l$
$\{\langle (a, b) : R \mid \alpha, -, - \rangle, \langle (a, b) : R \mid \beta, -, - \rangle\} \subseteq \mathcal{A}_i^{\mathcal{E}}$, with $\otimes(\alpha, \beta) = l$

The purpose of the clash triggers is to detect possible inconsistencies in the knowledge base. Note that the last two clash triggers detect the contradiction in terms of the certainty values specified for the same assertion. For example, suppose the certainty domain is $\mathcal{V} = \mathcal{C}[0, 1]$, i.e., the set of closed subintervals $[\alpha, \beta]$ in [0, 1] where $\alpha \preceq \beta$. If a knowledge base contains both assertions $\langle John : Tall \mid [0, 0.2], -, - \rangle$ and $\langle John : Tall \mid [0.7, 1], -, - \rangle$, then the third clash trigger will detect this as an inconsistency. Note that this clash triggers detects inconsistencies for atomic concepts. The contradictions in complex concepts are left to be detected by the constraint solver.

Concept Assertion Rule:

Condition:
$\langle a : A \mid \Gamma, -, - \rangle \in \mathcal{A}_i^{\mathcal{E}}$
Action:
if Γ is not the variable $x_{a:A}$
then $\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(x_{a:A} = \Gamma)\}$
$\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(x_{a:\neg A} = \sim\Gamma)\}$

This rule simply adds the certainty value of each atomic concept assertion and its negation to the constraints set $\mathcal{C}_j$. For example, suppose we have the assertion $\langle John : Tall \mid [0.6, 1], -, - \rangle$ in the extended ABox. If the certainty domain is $\mathcal{V} = \mathcal{C}[0, 1]$ and if the negation function is $\sim(x) = t - x$, where t is the top certainty in the lattice, then we add the constraints $(x_{John:Tall} = [0.6, 1])$ and $(x_{John:\neg Tall} = [0, 0.4])$ to the constraints set $\mathcal{C}_j$. On the other hand, if we have the assertion $\langle John : Tall \mid x_{John:\ Tall}, -, - \rangle$ in the extended ABox, we add the constraint $(x_{John:\neg Tall} = t - x_{John:Tall})$ to $\mathcal{C}_j$.

Role Assertion Rule:

Condition:
$\langle (a, b) : R \mid \Gamma, -, - \rangle \in \mathcal{A}_i^{\mathcal{E}}$
Action:
if Γ is not the variable $x_{(a,b):R}$
then $\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(x_{(a,b):R} = \Gamma)\}$

$\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(x_{\neg(a,b):R} = \sim\Gamma)\}$

Similar to the Concept Assertion Rule, this rule simply adds the certainty value of each atomic role assertion and its negation to the constraints set $\mathcal{C}_j$. For example, suppose we have the assertion $\langle (John, Diabetes) : hasDisease \mid 0.9, -, - \rangle$ in the extended ABox. If the certainty domain is $\mathcal{V} = [0, 1]$ and if the negation function is $\sim(x) = t - x$ where t is the top certainty in the lattice, then we add the constraints $(x_{(John,Diabetes):hasDisease} = 0.9)$ and $(x_{(John,Diabetes):\neg hasDisease} = 0.1)$ to $\mathcal{C}_j$. On the other hand, if the assertion $\langle (John, Diabetes) : hasDisease \mid x_{(John,Diabetes):hasDisease}, -, - \rangle$ is in the ABox, then the constraint $(x_{(John,Diabetes):\neg hasDisease} = t - x_{(John,Diabetes):hasDisease})$ is added to $\mathcal{C}_j$.

Negation Rule:

Condition:

$\langle a : \neg A \mid \Gamma, -, - \rangle \in \mathcal{A}_i^{\mathcal{E}}$

Action:

$\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle a : A \mid \sim\Gamma, -, - \rangle\}$

The intuition behind the Negation Rule is that, if we know an assertion has certainty value Γ, then the certainty of its negation can be obtained by applying the negation operator in the lattice to Γ. For example, suppose the certainty domain is $\mathcal{V} = [0, 1]$, and the negation operator is defined as $\sim(x) = 1 - x$. Then, if the assertion $\langle John : \neg Tall \mid 0.8, -, - \rangle$ is in the ABox, we could infer $\langle John : Tall \mid 0.2, -, - \rangle$, which is added to the extended ABox.

Conjunction Rule:

Condition:

$\langle a : C \sqcap D \mid \Gamma, f_c, f_d \rangle \in \mathcal{A}_i^{\mathcal{E}}$

Action:

for each $\Psi \in \{C, D\}$
if Ψ is atomic
then $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle a : \Psi \mid x_{a:\Psi}, -, - \rangle\}$
else $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle a : \Psi \mid x_{a:\Psi}, f_c, f_d \rangle\}$
$\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(f_c(x_{a:C}, x_{a:D}) = \Gamma)\}$

The intuition behind this rule is that, if we know an individual is in $C \sqcap D$, then we know it is in both C and D. In addition, according to the semantics of the description language, we know that the semantics of $a : C \sqcap D$ is defined by applying the conjunction function to the interpretation of $a : C$ and the interpretation of $a : D$.

For example, if the extended ABox includes the assertion $\langle Mary : Tall \sqcap Thin \mid 0.8, min, max \rangle$, then we could infer that $\langle Mary : Tall \mid x_{Mary:Tall}, -, - \rangle$ and $\langle Mary : Thin \mid x_{Mary:Thin}, -, - \rangle$. Also, the constraint $min(x_{Mary:Tall}, x_{Mary:Thin})$

= 0.8 must be satisfied.

Disjunction Rule:

Condition:

$\langle a : C \sqcup D \mid \Gamma, f_c, f_d \rangle \in \mathcal{A}_i^{\mathcal{E}}$

Action:

for each $\Psi \in \{C, D\}$
if Ψ is atomic
then $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle a : \Psi \mid x_{a:\Psi}, -, - \rangle\}$
else $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle a : \Psi \mid x_{a:\Psi}, f_c, f_d \rangle\}$
$\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(f_d(x_{a:C}, x_{a:D}) = \Gamma)\}$

The intuition behind this rule is that, if we know an individual is in $C \sqcup D$, then we know it is in either C, D, or in both. In addition, according to the semantics of the description language, we know that the semantics of $a : C \sqcup D$ is defined by applying the disjunction function to the interpretation of $a : C$ and that of $a : D$.

It is interesting to note is that the disjunction rule in the standard DL is non-deterministic, since it can be applied in different ways to the same ABox. However, note that the disjunction rule in $\mathcal{ALC}_U$ is deterministic. This is because the semantics of the concept disjunction is now encoded in the disjunction function in the form of a constraint. For example, suppose the extended ABox includes the assertion $\langle Mary : Tall \sqcup Thin \mid 0.8, min, max \rangle$, then we know that Mary is Tall to some degree ($\langle Mary : Tall \mid x_{Mary:Tall}, -, - \rangle$) and Mary is Thin to some degree ($\langle Mary : Thin \mid x_{Mary:Thin}, -, - \rangle$), possibly zero. Moreover, the constraint $max(x_{Mary:Tall}, x_{Mary:Thin}) = 0.8$ must be satisfied, which means that either $x_{Mary:Tall} = 0.8$, or $x_{Mary:Thin} = 0.8$, or $x_{Mary:Tall} = x_{Mary:Thin} = 0.8$.

Role Exists Restriction Rule:

Condition:

$\langle a : \exists R.C \mid \Gamma, f_c, f_d \rangle \in \mathcal{A}_i^{\mathcal{E}}$ and a is not blocked

Action:

if $\nexists$ individual b such that $(f_c(x_{(a,b):R}, x_{b:C}) = x_{a:\exists R.C}) \in \mathcal{C}_j$
then let b be a new individual
  $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle (a, b) : R \mid x_{(a,b):R}, -, - \rangle\}$
  if C is atomic
  then $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle b : C \mid x_{b:C}, -, - \rangle\}$
  else $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle b : C \mid x_{b:C}, f_c, f_d \rangle\}$
  $\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(f_c(x_{(a,b):R}, x_{b:C}) = x_{a:\exists R.C})\}$
  for each axiom $\langle \top \sqsubseteq D \mid \alpha, f_c, f_d \rangle$ in the TBox $\mathcal{T}$
    $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle b : D \mid \alpha, f_c, f_d \rangle\}$
if Γ is not the variable $x_{a:\exists R.C}$
then if $(x_{a:\exists R.C} = \Gamma') \in \mathcal{C}_j$

then if $\Gamma \neq \Gamma'$ and Γ is not an element in Γ'
then $\mathcal{C}_{j+1} = \mathcal{C}_j \setminus \{(x_{a:\exists R.C} = \Gamma')\} \cup \{(x_{a:\exists R.C} = \oplus(\Gamma, \Gamma'))\}$
else $\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(x_{a:\exists R.C} = \Gamma)\}$

The intuition behind this rule is that, if we know that an individual a is in $\exists R.C$, there must exist at least an individual, say b, such that a is related to b through the relationship R, and b is in the concept C. If no such individual b exists in the extended ABox, then we create such a new individual. In addition, this new individual must satisfy all the axioms in the TBox. For example, suppose the assertion $\langle \mathit{Tom} : \exists \mathit{hasDisease}.\mathit{Diabetes} \mid [0.4, 0.6], \mathit{min}, \mathit{max}\rangle$ is in the extended ABox and the axiom $\langle \top \sqsubseteq \neg \mathit{Obese} \sqcup \exists \mathit{hasDisease}.\mathit{Diabetes} \mid [0.7, 1], \times, \mathit{ind}\rangle$ is in the TBox. Assume that the ABox originally does not contain any individual b such that Tom is related to b through the role $\mathit{hasDisease}$, and b is in the concept $\mathit{Diabetes}$. Then, we could infer $\langle(\mathit{Tom}, d_1) : \mathit{hasDisease} \mid x_{(\mathit{Tom}, d_1):\mathit{hasDisease}}, -, -\rangle$ and $\langle d_1 : \mathit{Diabetes} \mid x_{d_1:\mathit{Diabetes}}, -, -\rangle$, where d_1 is a new individual. In addition, since d_1 must satisfy the axioms in the TBox, the assertion $\langle d_1 : \neg \mathit{Obese} \sqcup \exists \mathit{hasDisease}.\ \mathit{Diabetes} \mid [0.7, 1], \times, \mathit{ind}\rangle$ is added to the extended ABox. Finally, the constraints $(\mathit{min}(x_{(\mathit{Tom},d_1):\mathit{hasDisease}}, x_{d_1:\mathit{Diabetes}}) = x_{\mathit{Tom}:\exists \mathit{hasDisease}.\mathit{Diabetes}})$ as well as $(x_{\mathit{Tom}:\exists \mathit{hasDisease}.\mathit{Diabetes}} = [0.4, 0.6])$ must be satisfied. Now, suppose there is another assertion $\langle \mathit{Tom} : \exists \mathit{hasDisease}.\mathit{Diabetes} \mid [0.5, 0.9], \mathit{min}, \mathit{max}\rangle$ in the extended ABox. Then, when we apply the Role Exists Restriction Rule, we do not generate a new individual. Instead, we simply replace the constraint $(x_{\mathit{Tom}:\exists \mathit{hasDisease}.\mathit{Diabetes}} = [0.4, 0.6])$ in $\mathcal{C}_j$ with the constraint $(x_{\mathit{Tom}:\exists \mathit{hasDisease}.\mathit{Diabetes}} = \mathit{sup}([0.5, 0.9], [0.4, 0.6]))$, where sup is the join operator in the lattice $\oplus$. This new constraint takes into account the certainty value of the current assertion as well as that of the previous assertion.

Role Value Restriction Rule:

Condition:
$\{\langle a : \forall R.C \mid \Gamma, f_c, f_d\rangle, \langle(a, b) : R \mid \Gamma', -, -\rangle\} \subseteq \mathcal{A}_i^{\mathcal{E}}$
Action:
if C is atomic
then $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle b : C \mid x_{b:C}, -, -\rangle\}$
else $\mathcal{A}_{i+1}^{\mathcal{E}} = \mathcal{A}_i^{\mathcal{E}} \cup \{\langle b : C \mid x_{b:C}, f_c, f_d\rangle\}$
$\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(f_d(x_{\neg(a,b):R}, x_{b:c}) = x_{a:\forall R.C})\}$
if Γ is not the variable $x_{a:\forall R.C}$
then if $(x_{a:\forall R.C} = \Gamma'') \in \mathcal{C}_j$
then if $\Gamma \neq \Gamma''$ and Γ is not an element in Γ''
then $\mathcal{C}_{j+1} = \mathcal{C}_j \setminus \{(x_{a:\forall R.C} = \Gamma'')\} \cup \{(x_{a:\forall R.C} = \otimes(\Gamma, \Gamma''))\}$
else $\mathcal{C}_{j+1} = \mathcal{C}_j \cup \{(x_{a:\forall R.C} = \Gamma)\}$

The intuition behind the Role Value Restriction rule is that, if we know that an individual a is in $\forall R.C$, and if there is an individual b such that a is related to b through the relationship R, then b must be in the concept C. For example, assume we have assertions $\langle Jim : \forall hasPet.Dog \mid [0.4, 0.6], min, max\rangle$ and $\langle (Jim, d_1) : hasPet \mid [0.5, 0.8], -, -\rangle$ in the extended ABox. Then, we could infer $\langle d_1 : Dog \mid x_{d_1:Dog}, -, -\rangle$. In addition, the constraints $(max(x_{(Jim,d_1):\neg hasPet}, x_{d_1:Dog}) = x_{Jim:\forall hasPet.Dog})$ as well as $(x_{Jim:\forall hasPet.Dog} = [0.4, 0.6])$ must be satisfied. Now, suppose we have another assertion $\langle Jim : \forall hasPet.Dog \mid [0.5, 0.9], min, max\rangle$ in the extended ABox. Then, when we apply the Role Value Restriction rule, we simply replace the constraint $(x_{Jim:\forall hasPet.Dog} = [0.4,\ 0.6])$ in $\mathcal{C}_j$ with the constraint $(x_{Jim:\forall hasPet.Dog} = inf([0.5, 0.9], [0.4, 0.6]))$, where inf is the meet operator in the lattice $\otimes$. Note that the new constraint takes into account the certainty value of the current assertion as well as that of the previous assertion.

3.3.4 Correctness of the $\mathcal{ALC}_U$ Tableau Algorithm

We establish the correctness of the $\mathcal{ALC}_U$ tableau algorithm by showing that it is sound, complete, and terminates, as follows.

Lemma 8 ***(Soundness)*** *Let $\mathcal{A}^{\mathcal{E}'}$ be an extended ABox obtained from the extended ABox $\mathcal{A}^{\mathcal{E}}$ after applying the completion rule. Let $\mathcal{I}$ be an interpretation and π be an evaluation. Then, $\langle \mathcal{I}, \pi\rangle$ is a model of $\mathcal{A}^{\mathcal{E}}$ iff $\langle \mathcal{I}, \pi\rangle$ is a model of $\mathcal{A}^{\mathcal{E}'}$.*

Proof. The "if" direction: Let $\mathcal{C}$ be the constraints set associated with the extended ABox $\mathcal{A}^{\mathcal{E}}$, and $\mathcal{C}'$ be the constraints set associated with the extended ABox $\mathcal{A}^{\mathcal{E}'}$. Since $\mathcal{A}^{\mathcal{E}} \subseteq \mathcal{A}^{\mathcal{E}'}$ and $\mathcal{C} \subseteq \mathcal{C}'$, if $\langle \mathcal{I}, \pi\rangle$ is a model of $\mathcal{A}^{\mathcal{E}'}$, it is also a model of $\mathcal{A}^{\mathcal{E}}$.

The "only if" direction: We prove the claim by considering each completion rule. Since the cases of Concept Assertion, Role Assertion, and Negation rules are straightforward, we skip them here.

Let C and D be concepts, a and b be individuals in the domain, and R be a role. Also, let $\langle \mathcal{I}, \pi\rangle$ be a model of $\mathcal{A}^{\mathcal{E}}$, and assume that the following completion rule is triggered.

Conjunction Rule: By applying the conjunction rule to $\langle a : C \sqcap D \mid \Gamma, f_c, f_d\rangle$ in $\mathcal{A}^{\mathcal{E}}$, we obtain the extended ABox $\mathcal{A}^{\mathcal{E}'} = \mathcal{A}^{\mathcal{E}} \cup \{\langle a : C \mid x_{a:C}, f_c, f_d\rangle, \langle a : D \mid x_{a:D}, f_c, f_d\rangle\}$ and the constraints set $\mathcal{C}' = \mathcal{C} \cup \{(f_c(x_{a:C}, x_{a:D}) = \Gamma)\}$. Since $\langle \mathcal{I}, \pi\rangle$ is a model of $\mathcal{A}^{\mathcal{E}}$, $\mathcal{I}$ satisfies $\langle a : C \sqcap D \mid \Gamma, f_c, f_d\rangle$, and we know that, by definition, $(C \sqcap D)^{\mathcal{I}}(a) = f_c(C^{\mathcal{I}}(a), D^{\mathcal{I}}(a)) = \Gamma$. Therefore, the pair $(C^{\mathcal{I}}(a), D^{\mathcal{I}}(a))$ is in $\{(x, y) \mid f_c(x, y) = \Gamma\}$. Hence, there exists some $\alpha_1, \alpha_2 \in \mathcal{V}$ such that $C^{\mathcal{I}}(a) = \alpha_1$ and $D^{\mathcal{I}}(a) = \alpha_2$. That is, $\mathcal{I}$ satisfies both $\langle a : C \mid \alpha_1, f_c, f_d\rangle$ and $\langle a : D \mid \alpha_2, f_c, f_d\rangle$.

Disjunction Rule: The application of the disjunction rule to $\langle a : C \sqcup D \mid \Gamma, f_c, f_d\rangle$ in $\mathcal{A}^{\mathcal{E}}$ yields the extended ABox $\mathcal{A}^{\mathcal{E}'} = \mathcal{A}^{\mathcal{E}} \cup \{\langle a : C \mid x_{a:C}, f_c, f_d\rangle, \langle a : D \mid x_{a:D}, f_c, f_d\rangle\}$ and the constraints set $\mathcal{C}' = \mathcal{C} \cup \{(f_d(x_{a:C}, x_{a:D}) = \Gamma)\}$. Since $\langle \mathcal{I}, \pi\rangle$ is a model of $\mathcal{A}^{\mathcal{E}}$, $\mathcal{I}$ satisfies $\langle a : C \sqcup D \mid \Gamma, f_c, f_d\rangle$, and we know that, by definition, $(C \sqcup D)^{\mathcal{I}}(a) = f_d(C^{\mathcal{I}}(a), D^{\mathcal{I}}(a)) = \Gamma$. Therefore, the pair $(C^{\mathcal{I}}(a), D^{\mathcal{I}}(a))$ is in $\{(x, y) \mid f_d(x, y) = \Gamma\}$. Hence, there exists some $\alpha_1, \alpha_2 \in \mathcal{V}$ such that $C^{\mathcal{I}}(a) = \alpha_1$ and $D^{\mathcal{I}}(a) = \alpha_2$. That is, $\mathcal{I}$ satisfies both $\langle a : C \mid \alpha_1, f_c, f_d\rangle$ and $\langle a : D \mid \alpha_2, f_c, f_d\rangle$.

Role Exists Restriction Rule: When the role exists restriction rule is applied to $\langle a : \exists R.C \mid \Gamma, f_c, f_d\rangle$ in $\mathcal{A}^{\mathcal{E}}$, there are two possible augmentations to the extended ABox/constraints set: (i) There is already an individual b such that $\{\langle (a, b) : R \mid x_{(a,b):R}, -, -\rangle, \langle b : C \mid x_{b:C}, f_c, f_d\rangle\} \subseteq \mathcal{A}_c$ and $\{(f_c(x_{(a,b):R}, x_{b:C}) = x_{a:\exists R.C}), (x_{a:\exists R.C} = \Gamma')\} \subseteq \mathcal{C}$. In this case, we replace the constraint $(x_{a:\exists R.C} = \Gamma')$ with $(x_{a:\exists R.C} = \oplus(\Gamma, \Gamma'))$. (ii) A new individual b is generated, and we have $\mathcal{A}^{\mathcal{E}'} = \mathcal{A}^{\mathcal{E}} \cup \{\langle (a, b) : R \mid x_{(a,b):R}, -, -\rangle, \langle b : C \mid x_{b:C}, f_c, f_d\rangle\}$ as well as $\mathcal{C}' = \mathcal{C} \cup \{(f_c(x_{(a,b):R}, x_{b:C}) = x_{a:\exists R.C}), (x_{a:\exists R.C} = \Gamma)\}$. Since $\langle \mathcal{I}, \pi\rangle$ is a model of $\mathcal{A}^{\mathcal{E}}$, we know that $\mathcal{I}$ satisfies $\langle a : \exists R.C \mid \Gamma, f_c, f_d\rangle$, the evaluation π gives the certainty that a is in $\exists R.C$ (denoted $\pi(x_{a:\exists R.C})$), and by definition, we know that $(\exists R.C)^{\mathcal{I}}(a) = \oplus_{b\in\Delta^{\mathcal{I}}}\{f_c(R^{\mathcal{I}}(a, b), C^{\mathcal{I}}(b))\} = \pi(x_{a:\exists R.C})$. Hence, there are certainty values $\alpha_1, \alpha_2 \in \mathcal{V}$ such that $R^{\mathcal{I}}(a, b) = \alpha_1$ and $C^{\mathcal{I}}(b) = \alpha_2$. That is, $\mathcal{I}$ satisfies both $\langle (a, b) : R \mid \alpha_1, -, -\rangle$ and $\langle b : C \mid \alpha_2, f_c, f_d\rangle$.

Role Value Restriction Rule: Assume that the role value restriction rule is applied to $\langle a : \forall R.C \mid \Gamma, f_c, f_d\rangle$ in $\mathcal{A}^{\mathcal{E}}$. Then, for every individual b that is a R-successor of individual a, we either obtain the extended ABox $\mathcal{A}^{\mathcal{E}'} = \mathcal{A}^{\mathcal{E}} \cup \{\langle b : C \mid x_{b:C}, f_c, f_d\rangle\}$ and the constraints set $\mathcal{C}' = \mathcal{C} \cup \{(f_d(x_{\neg(a,b):R}, x_{b:c}) = x_{a:\forall R.C}), (x_{a:\forall R.C} = \Gamma)\}$, or in case the constraint $(x_{a:\forall R.C} = \Gamma'')$ is already in $\mathcal{C}$, we replace it with $(x_{a:\forall R.C} = \otimes(\Gamma, \Gamma''))$. Since $\langle \mathcal{I}, \pi\rangle$ is a model of $\mathcal{A}^{\mathcal{E}}$, $\mathcal{I}$ satisfies $\langle a : \forall R.C \mid \Gamma, f_c, f_d\rangle$ and, for every individual b that is a R-successor of a, $\mathcal{I}$ satisfies $\langle (a, b) : R \mid \Gamma', -, -\rangle$. In addition, the evaluation π gives the certainty that a is in $\forall R.C$ (denoted $\pi(x_{a:\forall R.C})$) and the certainty that b is a R-successor of a (denoted $\pi(x_{(a,b):R})$). We know by definition that $(\forall R.C)^{\mathcal{I}}(a) = \otimes_{b\in\Delta^{\mathcal{I}}}\{f_d(\sim R^{\mathcal{I}}(a, b), C^{\mathcal{I}}(b))\} = \pi(x_{a:\forall R.C})$. Hence, for every b that is a R-successor of a, there exists some $\alpha \in \mathcal{V}$ such that $R^{\mathcal{I}}(a, b) = \pi(x_{(a,b):R})$ and $C^{\mathcal{I}}(b) = \alpha$. That is, $\mathcal{I}$ satisfies $\langle b : C \mid \alpha, f_c, f_d\rangle$.

□

Lemma 9 ***(Completeness)*** *Any complete extended ABox $\mathcal{A}^{\mathcal{E}}_c$ has a model.*

Proof. Let $\mathcal{A}^{\mathcal{E}}_c$ be a complete extended ABox, and $\mathcal{C}$ be the constraints set associated with $\mathcal{A}^{\mathcal{E}}_c$. Since $\mathcal{A}^{\mathcal{E}}_c$ is complete, there exists an evaluation $\pi : Var(\mathcal{C}) \to \mathcal{V}$ that is a solution to the constraints set $\mathcal{C}$, where $Var(\mathcal{C})$ is the set of variables occurring in $\mathcal{C}$, and $\mathcal{V}$ is the certainty domain.

We now define a canonical interpretation $\mathcal{I}_{\mathcal{A}}$ of $\mathcal{A}_c^{\mathcal{E}}$ as follows:

- The domain $\Delta^{\mathcal{I}_{\mathcal{A}}}$ of $\mathcal{I}_{\mathcal{A}}$ consists of all the individual names occurring in $\mathcal{A}_c^{\mathcal{E}}$.
- For every atomic concept A in $\mathcal{A}_c^{\mathcal{E}}$, we define

$$A^{\mathcal{I}_{\mathcal{A}}}(a) = \begin{cases} \pi(x_{a:A}) & \text{if } \langle a : A \mid x_{a:A}, -, - \rangle \in \mathcal{A}_c^{\mathcal{E}} \\ \alpha & \text{if } \langle a : A \mid \alpha, -, - \rangle \in \mathcal{A}_c^{\mathcal{E}} \\ b & \text{otherwise, where } b \text{ is the least value in } \mathcal{V} \end{cases}$$

- For all roles R, we define

$$R^{\mathcal{I}_{\mathcal{A}}}(a_1, a_2) = \begin{cases} \pi(x_{(a_1,a_2):R}) & \text{if } \langle (a_1, a_2) : R \mid x_{(a_1,a_2):R}, -, - \rangle \in \mathcal{A}_c^{\mathcal{E}} \\ \alpha & \text{if } \langle (a_1, a_2) : R \mid \alpha, -, - \rangle \in \mathcal{A}_c^{\mathcal{E}} \\ b & \text{otherwise} \end{cases}$$

Next, we show that the pair $\langle \mathcal{I}_{\mathcal{A}}, \pi \rangle$ is a model of $\mathcal{A}_c^{\mathcal{E}}$. That is, $\mathcal{I}_{\mathcal{A}}$ satisfies all the assertions in $\mathcal{A}_c^{\mathcal{E}}$, and π is a solution to the constraints set $\mathcal{C}$.

By definition, $\mathcal{I}_{\mathcal{A}}$ satisfies all the role assertions in $\mathcal{A}_c^{\mathcal{E}}$. We now show that $\mathcal{I}_{\mathcal{A}}$ also satisfies all the concept assertions of the form $\langle a : C \mid \Gamma, f_c, f_d \rangle$ in $\mathcal{A}_c^{\mathcal{E}}$. For this, we use the induction technique on the structure of the concept C, where Γ is either a certainty value in the certainty domain or the variable $x_{a:C}$ denoting the certainty of the assertion.

Base Case:

If C is an atomic concept, then $\mathcal{I}_{\mathcal{A}}$ satisfies the concept assertion by definition.

Induction Step:

If $C = C_1 \sqcap C_2$, we have $\langle a : C_1 \sqcap C_2 \mid \Gamma, f_c, f_d \rangle \in \mathcal{A}_c^{\mathcal{E}}$. Since $\mathcal{A}_c^{\mathcal{E}}$ is complete, no more rule is applicable, and we have $\{\langle a : C_1 \mid x_{a:C_1}, f_c, f_d \rangle, \langle a : C_2 \mid x_{a:C_2}, f_c, f_d \rangle\} \subseteq \mathcal{A}_c^{\mathcal{E}}$ and $(f_c(x_{a:C_1}, x_{a:C_2}) = \Gamma) \in \mathcal{C}$. By the induction hypothesis, we know that $\mathcal{I}_{\mathcal{A}}$ satisfies $\langle a : C_1 \mid x_{a:C_1}, f_c, f_d \rangle$ and $\mathcal{I}_{\mathcal{A}}$ satisfies $\langle a : C_2 \mid x_{a:C_2}, f_c, f_d \rangle$. Also, since π is a solution to the constraints set $\mathcal{C}$, we have $f_c(\pi(x_{a:C_1}), \pi(x_{a:C_2})) = \Gamma$, where the evaluation π gives the certainties to variables $x_{a:C_1}$ and $x_{a:C_2}$. Hence, $f_c(C_1^{\mathcal{I}_{\mathcal{A}}}(a), C_2^{\mathcal{I}_{\mathcal{A}}}(a)) = \Gamma$. Since, according to Table 1, we have that $f_c(C_1^{\mathcal{I}}(a), C_2^{\mathcal{I}}(a)) = (C_1 \sqcap C_2)^{\mathcal{I}}(a)$, and since an interpretation $\mathcal{I}$ satisfies $\langle a : C_1 \sqcap C_2 \mid \Gamma, f_c, f_d \rangle$ if $(C_1 \sqcap C_2)^{\mathcal{I}}(a) = \Gamma$, the canonical interpretation $\mathcal{I}_{\mathcal{A}}$ satisfies concept assertions of the form $\langle a : C_1 \sqcap C_2 \mid \Gamma, f_c, f_d \rangle$.

If $C = C_1 \sqcup C_2$, we have $\langle a : C_1 \sqcup C_2 \mid \Gamma, f_c, f_d\rangle \in \mathcal{A}_c^{\mathcal{E}}$. Since $\mathcal{A}_c^{\mathcal{E}}$ is complete, no more rule is applicable, and we have $\{\langle a : C_1 \mid x_{a:C_1}, f_c, f_d\rangle, \langle a : C_2 \mid x_{a:C_2}, f_c, f_d\rangle\} \subseteq \mathcal{A}_c^{\mathcal{E}}$ and $(f_d(x_{a:C_1}, x_{a:C_2}) = \Gamma) \in \mathcal{C}$. By the induction hypothesis, we know that $\mathcal{I}_{\mathcal{A}}$ satisfies $\langle a : C_1 \mid x_{a:C_1}, f_c, f_d\rangle$ and $\mathcal{I}_{\mathcal{A}}$ satisfies $\langle a : C_2 \mid x_{a:C_2}, f_c, f_d\rangle$. Also, since π is a solution to the constraints set $\mathcal{C}$, we have $f_d(\pi(x_{a:C_1}), \pi(x_{a:C_2})) = \Gamma$. Hence, $f_d(C_1^{\mathcal{I}_{\mathcal{A}}}(a), C_2^{\mathcal{I}_{\mathcal{A}}}(a)) = \Gamma$. Since, according to Table 1, $f_d(C_1^{\mathcal{I}}(a), C_2^{\mathcal{I}}(a)) = (C_1 \sqcup C_2)^{\mathcal{I}}(a)$, and since an interpretation $\mathcal{I}$ satisfies $\langle a : C_1 \sqcup C_2 \mid \Gamma, f_c, f_d\rangle$ if $(C_1 \sqcup C_2)^{\mathcal{I}}(a) = \Gamma$, the canonical interpretation $\mathcal{I}_{\mathcal{A}}$ satisfies concept assertions of the form $\langle a : C_1 \sqcup C_2 \mid \Gamma, f_c, f_d\rangle$. Note that the proof presented here is much simpler than that of standard $\mathcal{ALC}$. This is due to the fact that the disjunction rule in $\mathcal{ALC}$ is nondeterministic, while the disjunction rule in $\mathcal{ALC}_U$ is deterministic, as we explained in Section 3.3.3.

If $C = \neg A$, we have $\langle a : \neg A \mid \Gamma, -, -\rangle \in \mathcal{A}_c^{\mathcal{E}}$. Since $\mathcal{A}_c^{\mathcal{E}}$ is complete, no more rule is applicable, and we have $\langle a : A \mid \sim\Gamma, -, -\rangle \in \mathcal{A}_c^{\mathcal{E}}$ and $\{(x_{a:A} = \sim\Gamma), (x_{a:\neg A} = \Gamma)\} \subseteq \mathcal{C}$. Since π is a solution to the constraints set $\mathcal{C}$, we have $\pi(x_{a:\neg A}) = \Gamma$, where π gives the evaluation to the variable $x_{a:\neg A}$. Hence, $\sim A^{\mathcal{I}_{\mathcal{A}}}(a) = \Gamma$. Since $\sim A^{\mathcal{I}}(a) = (\neg A)^{\mathcal{I}}(a)$ according to Table 1, and since an interpretation $\mathcal{I}$ satisfies $\langle a : \neg A \mid \Gamma, -, -\rangle$ if $(\neg A)^{\mathcal{I}}(a) = \Gamma$, the canonical interpretation $\mathcal{I}_{\mathcal{A}}$ satisfies concept assertions of the form $\langle a : \neg A \mid \Gamma, -, -\rangle$.

If $C = \exists R.C_1$, we have $\langle a : \exists R.C_1 \mid \Gamma, f_c, f_d\rangle \in \mathcal{A}_c^{\mathcal{E}}$. Since $\mathcal{A}_c^{\mathcal{E}}$ is complete, no more rule is applicable. The application of the Role Exists Restriction rule either (i) generated a new individual b, added assertions $\{\langle (a, b) : R \mid x_{(a,b):R}, -, -\rangle, \langle b : C_1 \mid x_{b:C_1}, f_c, f_d\rangle\}$ to the extended ABox $\mathcal{A}_c^{\mathcal{E}}$, and added the constraint $(f_c(x_{(a,b):R}, x_{b:C_1}) = x_{a:\exists R.C_1})$ to the constraints set $\mathcal{C}$, or (ii) did not generate a new individual because there already existed an individual b such that $\{\langle (a, b) : R \mid x_{(a,b):R}, -, -\rangle, \langle b : C_1 \mid x_{b:C_1}, f_c, f_d\rangle\} \subseteq \mathcal{A}_c^{\mathcal{E}}$, and the constraint $(f_c (x_{(a,b):R}, x_{b:C_1}) = x_{a:\exists R.C_1})$ was already in the constraints set $\mathcal{C}$, or (iii) did not generate new individual because a is blocked by some ancestor b with $\mathcal{A}_i^{\mathcal{E}}(a) \subseteq \mathcal{A}_i^{\mathcal{E}}(b)$; in such case, we could construct the model by having $\langle (a, b) : R \mid x_{(a,b):R}, -, -\rangle$ and $\langle b : C_1 \mid x_{b:C_1}, f_c, f_d\rangle$. In all the three cases, there exists at least one individual b such that $\langle (a, b) : R \mid x_{(a,b):R}, -, -\rangle$, $\langle b : C_1 \mid x_{b:C_1}, f_c, f_d\rangle$, and $f_c(x_{(a,b):R}, x_{b:C_1}) = x_{a:\exists R.C_1}$. By the induction hypothesis, we know that for each individual b such that (a, b) is in R and b is in C_1, $\mathcal{I}_{\mathcal{A}}$ satisfies $\langle (a, b) : R \mid x_{(a,b):R}, -, -\rangle$ and $\langle b : C_1 \mid x_{b:C_1}, f_c, f_d\rangle$. Also, since π is a solution to constraints set $\mathcal{C}$, we have $\oplus_{b\in\Delta^{\mathcal{I}}}\{f_c(\pi(x_{(a,b):R}), \pi(x_{b:C_1}))\} = \pi(x_{a:\exists R.C_1})$. Hence, $\oplus_{b\in\Delta^{\mathcal{I}_{\mathcal{A}}}}\{f_c(R^{\mathcal{I}_{\mathcal{A}}}(a, b),\ C_1{}^{\mathcal{I}_{\mathcal{A}}}(b))\} = (\exists R.C_1)^{\mathcal{I}_{\mathcal{A}}}(a)$. That is, $\mathcal{I}_{\mathcal{A}}$ satisfies concept assertions of the form $\langle a : \exists R.C_1 \mid \Gamma, f_c, f_d\rangle$.

If $C = \forall R.C_1$, we have $\langle a : \forall R.C_1 \mid \Gamma, f_c, f_d\rangle \in \mathcal{A}_c^{\mathcal{E}}$. Since $\mathcal{A}_c^{\mathcal{E}}$ is complete, no more rule is applicable, and for every individual b such that $\langle (a, b) : R \mid x_{(a,b):R}, -, -\rangle \in \mathcal{A}_c^{\mathcal{E}}$, we have $\langle b : C_1 \mid x_{b:C_1}, f_c, f_d\rangle \in \mathcal{A}_c^{\mathcal{E}}$ and also $(f_d(x_{(a,b):\neg R}, x_{b:C_1}) = x_{a:\forall R.C_1}) \in \mathcal{C}$. By the induction hypothesis, we know that for each individual b such that (a, b) is in R and b is in C_1, $\mathcal{I}_{\mathcal{A}}$ satisfies $\langle (a, b) : R \mid x_{(a,b):R}, -, -\rangle$

and $\mathcal{I}_\mathcal{A}$ satisfies $\langle b : C_1 \mid x_{b:C_1}, f_c, f_d\rangle$. Also, since π is a solution to the constraints set $\mathcal{C}$, we have $\otimes_{b\in\Delta^\mathcal{I}}\{f_d(\pi(x_{(a,b):\neg R}), \pi(x_{b:C_1}))\} = \pi(x_{a:\forall R.C_1})$. Hence, $\otimes_{b\in\Delta^{\mathcal{I}_\mathcal{A}}}\{f_d(\sim R^{\mathcal{I}_\mathcal{A}}(a, b), {C_1}^{\mathcal{I}_\mathcal{A}}(b))\} = (\forall R.\ C_1)^{\mathcal{I}_\mathcal{A}}(a)$. That is, $\mathcal{I}_\mathcal{A}$ satisfies concept assertions of the form $\langle a : \forall R.C_1 \mid \Gamma, f_c, f_d\rangle$.

□

Lemma 10 *If an extended ABox $\mathcal{A}_c^\mathcal{E}$ contains a clash, or if the constraints set $\mathcal{C}$ associated with $\mathcal{A}_c^\mathcal{E}$ is unsolvable, then $\mathcal{A}_c^\mathcal{E}$ does not have a model.*

Proof. If an extended ABox $\mathcal{A}_c^\mathcal{E}$ contains a clash, then no interpretation can satisfy $\mathcal{A}_c^\mathcal{E}$. Thus, $\mathcal{A}_c^\mathcal{E}$ is inconsistent and has no model. Similarly, if the constraints set $\mathcal{C}$ associated with $\mathcal{A}_c^\mathcal{E}$ is unsolvable, there does not exist an evaluation $\pi : Var(\mathcal{C}) \to \mathcal{V}$ that is a solution to the constraints set $\mathcal{C}$, where $Var(\mathcal{C})$ is the set of variables occurring in the constraints set $\mathcal{C}$, and $\mathcal{V}$ is the certainty domain. Hence, $\mathcal{A}_c^\mathcal{E}$ is not satisfied and has no model.

□

Before proving the termination property, we need to introduce the term "concept subsets."

Definition 11 *(Concept Subsets) Let C be a concept. The subsets of C, denoted $subset(C)$, is recursively defined as follows.*

$subset(A) = \{A\}$, *where A is an atomic concept*
$subset(C_1 \sqcap C_2) = \{C_1 \sqcap C_2\} \cup subset(C_1) \cup subset(C_2)$
$subset(C_1 \sqcup C_2) = \{C_1 \sqcup C_2\} \cup subset(C_1) \cup subset(C_2)$
$subset(\exists R.C_1) = \{\exists R.C_1\} \cup subset(C_1)$
$subset(\forall R.C_1) = \{\forall R.C_1\} \cup subset(C_1)$

Lemma 12 ***(Termination)*** *Let X be any assertion in the extended ABox $\mathcal{A}^\mathcal{E}$. The application of the completion rules to X always terminates.*

Proof. Let X be of the form $\langle a : C \mid \alpha, f_c, f_d\rangle$, and $s = |subset(C)|$. As in the standard DL [19], termination is a result of the following properties of the completion rules:

(1) The completion rules are designed to avoid duplicated rule applications.
(2) The completion rules never remove any assertion from the extended ABox nor change/remove any concept in the assertion.

(3) Successors are only generated by Role Exists Restriction Rule, and each application of such rule generates at most one successor. Since there cannot be more than s Role Exists Restrictions, the out-degree of the tree is bounded by s.
(4) Each node label contains non-empty subsets of $subset(C)$. Hence, if there is a path of length at least 2^s, there must be two nodes along the path that have the same node label, and hence blocking occurs. Since the path cannot grow longer once a blocking takes place, the length of the path is at most 2^s.

□

3.4 Illustrative Examples

To illustrate the $\mathcal{ALC}_U$ framework, we first present a detailed example of how $\mathcal{ALC}_U$ tableau algorithm can be applied step-by-step. We then present an example demonstrating some interesting inferences that can be performed on an $\mathcal{ALC}_U$ knowledge base.

3.4.1 Example of Applying $\mathcal{ALC}_U$ Tableau Algorithm

To illustrate the $\mathcal{ALC}_U$ tableau algorithm and the need for blocking, let us consider a cyclic fuzzy knowledge base $\Sigma = \langle \mathcal{T}, \mathcal{A} \rangle$, where:

$$\mathcal{T} = \{\langle ObesePerson \sqsubseteq \exists hasParent.ObesePerson \mid [0.7, 1], min, max\rangle\}$$

$$\mathcal{A} = \{\langle John : ObesePerson \mid [0.8, 1], -, -\rangle\}$$

Note that the fuzzy knowledge bases can be expressed in $\mathcal{ALC}_U$ by setting the certainty lattice as $\mathcal{L} = \langle \mathcal{V}, \preceq \rangle$, where $\mathcal{V} = \mathcal{C}[0, 1]$ is the set of closed subintervals $[\alpha, \beta]$ in [0, 1] such that $\alpha \preceq \beta$. We also set the meet operator in the lattice as *inf* (infimum), the join operator as *sup* (supremum), and the negation function as $\sim(x) = t - x$, where $t = [1, 1]$ is the greatest value in the certainty lattice. Finally, the conjunction function is set to min, and the disjunction function is set to max.

To find out if Σ is consistent, we first apply the pre-processing steps. For this, we transform the axiom into its normal form:

$$\mathcal{T} = \{\langle \top \sqsubseteq (\neg ObesePerson \sqcup \exists hasParent.ObesePerson \mid [0.7, 1], min, max\rangle\}$$

We then augment the ABox with respect to the TBox. That is, for each individual a in the ABox (in this case, we have only $John$) and for each axiom of the form $\langle \top \sqsubseteq \neg C \sqcup D \mid \alpha, f_c, f_d \rangle$ in the TBox, we add an assertion $\langle a : \neg C \sqcup D \mid \alpha, f_c, f_d \rangle$ to the ABox. Hence, in this step, we add the following assertion to the ABox:

$$\langle John : (\neg ObesePerson \sqcup \exists hasParent.ObesePerson \mid [0.7, 1], min, max \rangle\}$$

Now, we can initialize the extended ABox to be:

$$\mathcal{A}_0^{\mathcal{E}} = \{\langle John : ObesePerson \mid [0.8, 1], -, - \rangle,$$
$$\langle John : (\neg ObesePerson \sqcup \exists hasParent.ObesePerson \mid [0.7, 1], min, max \rangle\}$$

and the constraints set to be $\mathcal{C}_0 = \{\}$.

Once the pre-processing phase is over, we are ready to apply the completion rules. The first assertion is $\langle John : ObesePerson \mid [0.8, 1], -, - \rangle$. Since *ObesePerson* is an atomic concept, we apply the Concept Assertion Rule, which yields:

$\mathcal{C}_1 = \mathcal{C}_0 \cup \{(x_{John:ObesePerson} = [0.8, 1])\}$
$\mathcal{C}_2 = \mathcal{C}_1 \cup \{(x_{John:\neg ObesePerson} = t - x_{John:ObesePerson})\}$, where t is the greatest element in the lattice, $[1, 1]$.

The other assertion in $\mathcal{A}_0^{\mathcal{E}}$ is $\langle John : (\neg ObesePerson \sqcup \exists hasParent.ObesePerson \mid [0.7, 1], min, max \rangle\}$. Since this assertion includes a concept disjunction, the Disjunction Rule applies. This yields:

$$\mathcal{A}_1^{\mathcal{E}} = \mathcal{A}_0^{\mathcal{E}} \cup \{\langle John : \neg ObesePerson \mid x_{John:\neg ObesePerson}, -, - \rangle\}$$
$$\mathcal{A}_2^{\mathcal{E}} = \mathcal{A}_1^{\mathcal{E}} \cup \{\langle John : \exists hasParent.ObesePerson \mid x_{John:\exists hasParent.ObesePerson}, min, max \rangle\}$$
$$\mathcal{C}_3 = \mathcal{C}_2 \cup \{(max(x_{John:\neg ObesePerson}, x_{John:\exists hasParent.ObesePerson}) = [0.7, 1])\}$$

The assertion $\langle John : \neg ObesePerson \mid x_{John:\neg ObesePerson}, -, - \rangle$ in $\mathcal{A}_1^{\mathcal{E}}$ triggers the Negation Rule, which yields:

$$\mathcal{A}_3^{\mathcal{E}} = \mathcal{A}_2^{\mathcal{E}} \cup \{\langle John : ObesePerson \mid x_{John:ObesePerson}, -, - \rangle\}$$

The application of the Concept Assertion Rule to the assertion $\langle John : ObesePerson \mid x_{John:ObesePerson}, -, - \rangle$ in $\mathcal{A}_3^{\mathcal{E}}$ does not derive any new assertion nor constraint. Next, we apply the Role Exists Restriction Rule to the assertion in $\mathcal{A}_2^{\mathcal{E}}$, and obtain:

$$\mathcal{A}_4^{\mathcal{E}} = \mathcal{A}_3^{\mathcal{E}} \cup \{\langle (John, ind1) : hasParent \mid x_{(John,ind1):hasParent}, -, - \rangle\}$$
$$\mathcal{A}_5^{\mathcal{E}} = \mathcal{A}_4^{\mathcal{E}} \cup \{\langle ind1 : ObesePerson \mid x_{ind1:ObesePerson}, -, - \rangle\}$$
$$\mathcal{C}_4 = \mathcal{C}_3 \cup \{(min(x_{(John,ind1):hasParent}, x_{ind1:ObesePerson}) = x_{John:\exists hasParent.ObesePerson})\}$$
$$\mathcal{A}_6^{\mathcal{E}} = \mathcal{A}_5^{\mathcal{E}} \cup \{\langle ind1 : (\neg ObesePerson \sqcup \exists hasParent.ObesePerson \mid [0.7, 1],$$

$min, max\rangle\}$

The application of the Role Assertion Rule to the assertion in $\mathcal{A}_4^{\mathcal{E}}$ yields:

$$\mathcal{C}_5 = \mathcal{C}_4 \cup \{(x_{(John,ind1):\neg hasParent} = t - x_{(John,ind1):hasParent})\}$$

After applying the Concept Assertion Rule to the assertion $\langle ind1 : ObesePer-son \mid x_{ind1:ObesePerson}, -, -\rangle\}$ in $\mathcal{A}_5^{\mathcal{E}}$, we obtain:

$$\mathcal{C}_6 = \mathcal{C}_5 \cup \{(x_{ind1:\neg ObesePerson} = t - x_{ind1:ObesePerson})\}$$

The assertion in $\mathcal{A}_6^{\mathcal{E}}$ triggers the Disjunction Rule, which yields:

$$\begin{aligned}
\mathcal{A}_7^{\mathcal{E}} &= \mathcal{A}_6^{\mathcal{E}} \cup \{\langle ind1 : \neg ObesePerson \mid x_{ind1:\neg ObesePerson}, -, -\rangle\} \\
\mathcal{A}_8^{\mathcal{E}} &= \mathcal{A}_7^{\mathcal{E}} \cup \{\langle ind1 : \exists hasParent.ObesePerson \mid x_{ind1:\exists hasParent.ObesePerson}, \\
&\quad min, max\rangle\} \\
\mathcal{C}_7 &= \mathcal{C}_6 \cup \{(max(x_{ind1:\neg ObesePerson}, x_{ind1:\exists hasParent.ObesePerson}) = [0.7, 1])\}
\end{aligned}$$

Next, the application of the Negation Rule to the assertion in $\mathcal{A}_7^{\mathcal{E}}$ yields:

$$\mathcal{A}_9^{\mathcal{E}} = \mathcal{A}_8^{\mathcal{E}} \cup \{\langle ind1 : ObesePerson \mid x_{ind1:ObesePerson}, -, -\rangle\}$$

We then apply the Concept Assertion Rule to the assertion in $\mathcal{A}_9^{\mathcal{E}}$, and obtain:

$$\mathcal{C}_8 = \mathcal{C}_7 \cup \{(x_{ind1:\neg ObesePerson} = t - x_{ind1:ObesePerson})\}$$

The application of the Role Exists Restriction Rule to the assertion in $\mathcal{A}_8^{\mathcal{E}}$ yields:

$$\begin{aligned}
\mathcal{A}_{10}^{\mathcal{E}} &= \mathcal{A}_9^{\mathcal{E}} \cup \{\langle (ind1, ind2) : hasParent \mid x_{(ind1,ind2):hasParent}, -, -\rangle\} \\
\mathcal{A}_{11}^{\mathcal{E}} &= \mathcal{A}_{10}^{\mathcal{E}} \cup \{\langle ind2 : ObesePerson \mid x_{ind2:ObesePerson}, -, -\rangle\} \\
\mathcal{C}_9 &= \mathcal{C}_8 \cup \{(min(x_{(ind1,ind2):hasParent}, x_{ind2:ObesePerson}) = x_{ind1:\exists hasParent.} \\
&\quad {}_{ObesePerson})\} \\
\mathcal{A}_{12}^{\mathcal{E}} &= \mathcal{A}_{11}^{\mathcal{E}} \cup \{\langle ind2 : (\neg ObesePerson \sqcup \exists hasParent.ObesePerson \mid [0.7, 1], \\
&\quad min, max\rangle\}
\end{aligned}$$

Next, the Role Assertion Rule is applied to the assertion in $\mathcal{A}_{10}^{\mathcal{E}}$ yields:

$$\mathcal{C}_{10} = \mathcal{C}_9 \cup \{(x_{(ind1,ind2):\neg hasParent} = t - x_{(ind1,ind2):hasParent})\}$$

After applying the Concept Assertion Rule to the assertion in $\mathcal{A}_{11}^{\mathcal{E}}$, we obtain:

$$\mathcal{C}_{11} = \mathcal{C}_{10} \cup \{(x_{ind2:\neg ObesePerson} = t - x_{ind2:ObesePerson})\}$$

The assertion in $\mathcal{A}_{12}^{\mathcal{E}}$ triggers the Disjunction Rule, which yields:

$$\begin{aligned}
\mathcal{A}_{13}^{\mathcal{E}} &= \mathcal{A}_{12}^{\mathcal{E}} \cup \{\langle ind2 : \neg ObesePerson \mid x_{ind2:\neg ObesePerson}, -, -\rangle\} \\
\mathcal{A}_{14}^{\mathcal{E}} &= \mathcal{A}_{13}^{\mathcal{E}} \cup \{\langle ind2 : \exists hasParent.ObesePerson \mid x_{ind2:\exists hasParent.ObesePerson},
\end{aligned}$$

$min, max\rangle\}$
$$\mathcal{C}_{12} = \mathcal{C}_{11} \cup \{(max(x_{ind2:\neg ObesePerson}, x_{ind2:\exists hasParent.ObesePerson}) = [0.7, 1])\}$$

Next, the application of the Negation Rule to the assertion in $\mathcal{A}^{\mathcal{E}}_{13}$ yields:

$$\mathcal{A}^{\mathcal{E}}_{15} = \mathcal{A}^{\mathcal{E}}_{14} \cup \{\langle ind2 : ObesePerson \mid x_{ind2:ObesePerson}, -, -\rangle\}$$

We then apply the Concept Assertion Rule to the assertion in $\mathcal{A}^{\mathcal{E}}_{15}$, and obtain:

$$\mathcal{C}_{13} = \mathcal{C}_{12} \cup \{(x_{ind2:\neg ObesePerson} = t - x_{ind2:ObesePerson})\}$$

Next, consider the assertion in $\mathcal{A}^{\mathcal{E}}_{14}$. Since $ind1$ is an ancestor of $ind2$ and $\mathcal{L}(ind2) \subseteq \mathcal{L}(ind1)$, individual $ind2$ is blocked. Therefore, we will not continue applying the Role Exists Restriction Rule to the assertion in $\mathcal{A}^{\mathcal{E}}_{14}$, and the completion rule application terminates at this point. Note that without blocking, the tableau algorithm would never terminate since new individual will be generated for each application of the Role Exists Restriction Rule.

Since there is no more rule applicable, the set of constraints in $\mathcal{C}_{13}$ is fed into the constraint solver to check its solvability. Since the constraints are solvable, the knowledge base is consistent.

3.4.2 Example of Reasoning with $\mathcal{ALC}_U$ knowledge base

In this section, we demonstrate some practical $\mathcal{ALC}_U$ queries by extending the example described in the Introduction. For simplicity, we use only min as the conjunction function, and max as the disjunction function.

The statement "The certainty that an obese person would have heart disease lies between 0.7 and 1" can be expressed in $\mathcal{ALC}_U$ as a fuzzy axiom $\langle ObesePerson \sqsubseteq HeartPatient \mid [0.7,\ 1], min, max\rangle$, and the statement "John is obese with a degree between 0.8 and 1" can be captured as an assertion $\langle John : ObesePerson \mid [0.8, 1], -, -\rangle$. Assume that, in addition to the above information, we also know that John is a male person ($\langle John : MalePerson \mid [1, 1], -, -\rangle$). His mother, Mary, is a diabetes patient ($\langle (John, Mary) : hasMother \mid [1, 1], -, -\rangle$ and $\langle Mary : DiabetesPatient \mid [1, 1], -, -\rangle$). We also know that the certainty of a female person being a breast cancer patient is at least 0.65 ($\langle FemalePerson \sqsubseteq BreastCancerPatient \mid [0.65,\ 1], min, max\rangle$), and the certainty that somebody who has a diabetes mother is a diabetes patient is at least 0.9 ($\langle \exists hasMother.DiabetesPatient \sqsubseteq DiabetesPatient \mid [0.9, 1], min, max\rangle$). Finally, we also know some general information, such as a male person is disjoint with a female person ($\langle MalePerson \sqcap FemalePerson \sqsubseteq \bot \mid [1, 1], min, max\rangle$), and that the range of the role hasMother is a female person ($\langle \top \sqsubseteq \forall hasMother.FemalePerson \mid [1, 1],\ min, max\rangle$).

With the above knowledge base (Σ), some interesting inferences can be performed. For example, to determine the certainty with which John is a heart patient, we apply the entailment checking by determining the degree that Σ entails the assertion $\langle John : HeartPatient \mid x_{John:HeartPatient}, min, max\rangle$, which yields a certainty between 0.7 and 1.

It is also interesting to find out the certainty that John is a diabetes patient or a heart patient by determining the degree that Σ entails the assertion $\langle John : (DiabetesPatient \sqcup HeartPatient) \mid x_{John:(DiabetesPatient \sqcup HeartPatient)}, min, max\rangle$, which yields a degree between 0.9 and 1. This is because the certainty with which John is a diabetes patient is at least 0.9, and the certainty that John is a heart patient is at least 0.7.

Finally, to see the certainty that John has a mother who is both a breast cancer patient and a diabetes patient, the entailment degree for the assertion $\langle John : \exists hasMother.(BreastCancerPatient \sqcap DiabetesPatient) \mid x_{John:\exists hasMother.(BreastCancerPatient \sqcap DiabetesPatient)}, min, max\rangle$ is determined, which yields a degree of at least 0.65. It is interesting to note that, although we did not explicitly assert that Mary is a female person, we can infer that Mary is a breast cancer patient with a certainty of at least 0.65 through the fact that Mary is John's mother, a mother is a female person, and a female person is a breast cancer patient with a certainty of at least 0.65.

4 Conclusion and Future Work

In this paper, we presented the $\mathcal{ALC}_U$ framework that extends the standard DL $\mathcal{ALC}$ with uncertainty. The proposed framework allows us to incorporate various forms of uncertainty within DLs in a uniform manner. This is achieved by abstracting away the notion of uncertainty in the description language, the knowledge base, and the reasoning procedure. The proposed tableau reasoning procedure works by deriving a set of assertions as well as linear/nonlinear constraints that encode the semantics of the uncertainty knowledge base. The advantage of this approach is that it makes the design of the $\mathcal{ALC}_U$ tableau algorithm generic and uniform for computing different semantics. That is, by simply tuning the combination functions that generate the constraints, different notions of uncertainty can be modeled and reasoned with, using a single reasoning procedure. To establish correctness of the $\mathcal{ALC}_U$ tableau algorithm, we showed that it is sound, complete, and terminates. We also demonstrated through examples that the $\mathcal{ALC}_U$ framework is capable of handling practical queries.

The optimization aspect of the $\mathcal{ALC}_U$ reasoning procedure is beyond the scope of this paper. However, a preliminary study in this regard can be found in [15].

As future work, we plan to extend $\mathcal{ALC}_U$ to support a more expressive portion of DL (e.g., $\mathcal{SHOIN}$, which OWL DL is based on) so that constructors such as number restrictions and transitive properties can be supported. Another interesting extension to the $\mathcal{ALC}_U$ framework would be to support other forms of uncertainty. Currently, we keep the description language syntax the same as the standard DL while extending only its semantics. However, since probabilistic reasoning usually requires extra information about the events, their relationships, and the facts in the world, it would require syntactical extension to the description language in order to model knowledge bases with more probability modes, such as positive/negative correlation and conditional probability. The challenge here would be investigating whether it is feasible to extend the syntax of the description language generically to support these uncertainty formalisms, and how such extension can fit into the existing $\mathcal{ALC}_U$ framework.

References

[1] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, P.F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation, and Applications, Cambridge University Press, Cambridge, UK, 2003.

[2] F. Baader, I. Horrocks, U. Sattler, Description logics for the semantic web, KI – Künstliche Intelligenz, No. 4, 2002, pp. 57-59.

[3] F. Baader, I. Horrocks, U. Sattler, Description logics, Handbook of Knowledge Representation. Elsevier, 2007.

[4] T. Berners-Lee, M. Fischetti, Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its Inventor, HarperCollins, New York, 2000, p. 177.

[5] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, Scientific American. 284(5) (2001) 34–43.

[6] P. Berry, K. Myers, T. Uribe, N. Yorke-Smith, Task management under change and uncertainty: constraint solving experience with the CALO project, Proceedings of CP'05 Workshop on Constraint Solving under Change and Uncertainty, Sitges, Spain, 2005.

[7] F. Bobillo, U. Straccia, A Fuzzy Description Logic with Product T-norm, Proceedings of the IEEE International Conference on Fuzzy Systems (2007), pp. 652-657.

[8] T. Copeland, Generating Parsers with JavaCC, Centennial Books, Alexandria, VA., 2007.

[9] M. Dürig, T. Studer, Probabilistic abox reasoning: preliminary results, Proceedings of the International Workshop on Description Logics (DL'05), 2005, pp. 104–111.

[10] E.C. Freuder, A.K. Mackworth (Eds.), Constraint-Based Reasoning, MIT Press, Cambridge, MA, USA, 1994.

[11] R. Giugno, T. Lukasiewicz, P-SHOQ(D): A probabilistic extension of SHOQ(D) for probabilistic ontologies in the semantic web, Proceedings of the European Conference on Logics in Artificial Intelligence, Cosenza, Italy, Springer Verlag, 2002, pp. 86–97.

[12] T.R. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition. 5(2) (1993) 199–220.

[13] V. Haarslev, H.I. Pai, N. Shiri, Uncertainty reasoning in description logics: a generic approach, Proceedings of the 19th International FLAIRS Conference, Melbourne Beach, Florida, AAAI Press, 2006, pp. 818–823.

[14] V. Haarslev, H.I. Pai, N. Shiri, Completion rules for uncertainty reasoning with the description logic $\mathcal{ALC}$, Proceedings of the Canadian Semantic Web Working Symposium, Quebec City, Canada, Series: Semantic Web and Beyond: Computing for Human Experience, Vol. 4, Springer Verlag, 2006, pp. 205–225.

[15] V. Haarslev, H.I. Pai, N. Shiri, Optimizing tableau reasoning in $\mathcal{ALC}$ extended with uncertainty, Proceedings of the International Workshop on Description Logics (DL'07), Brixen-Bressanone, Italy, 2007, pp. 307–314.

[16] V. Haarslev, H.I. Pai, N. Shiri, Semantic web uncertainty management, Encyclopedia of Information Science and Technology, 2nd edition, Information Science Reference, 2008.

[17] S. Hölldobler, T.D. Khang, H.P. Störr, A fuzzy description logic with hedges as concept modifiers, Proceedings of the 3rd International Conference on Intelligent Technologies, Hanoi, Vietnam, Science and Technics Publishing House, 2002, pp. 25–34.

[18] B. Hollunder, An alternative proof method for possibilistic logic and its application to terminological logics, Proceedings of the 10th Annual Conference on Uncertainty in Artificial Intelligence, San Francisco, CA, Morgan Kaufmann, 1994, pp. 327–335.

[19] I. Horrocks, U. Sattler, S. Tobies, A description logic with transitive and converse roles, role hierarchies and qualifying number restrictions, Tech. Report LTCS-Report 99-08, LuFg Theoretical Computer Science, RWTH Aachen, Germany, 1999.

[20] M. Jaeger, Probabilistic reasoning in terminological logics, Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning, 1994, pp. 305–316.

[21] D. Koller, A.Y. Levy, A. Pfeffer, P-CLASSIC: a tractable probablistic description logic, Proceedings of the 14th National Conference on Artificial Intelligence, Providence, Rhode Island, AAAI Press, 1997, pp. 390–397.

[22] L.V.S. Lakshmanan, N. Shiri, A parametric approach to deductive databases with uncertainty, IEEE Transactions on Knowledge and Data Engineering. 13(4) (2001) 554–570.

[23] L.V.S. Lakshmanan, N. Shiri, Logic programming and deductive databases with uncertainty: a survey, Enclyclopedia of Computer Science and Technology, Vol. 45, Marcel Dekker Inc., New York, 2001, pp. 153–176.

[24] T. Lukasiewicz, Probabilistic description logic programs, International Journal of Approximate Reasoning, 45(2) (2007) 288–307.

[25] T. Lukasiewicz, Probabilistic description logic programs under inheritance with overriding for the Semantic Web, International Journal of Approximate Reasoning, 49(1) (2008) 18–34.

[26] A. Motro, Ph. Smets (Eds.), Uncertainty Management in Information Systems: From Needs to Solutions, Kluwer Academic Publishers, Boston, 1997.

[27] G. Qi, J.Z. Pan, Q. Ji, Possibilistic extension of description logics, Proceedings of the International Workshop on Description Logics (DL'07), 2007, pp. 435–442.

[28] D. Sánchez, A.G.B. Tettamanzi, Generalizing quantification in fuzzy description logics, Proceedings of the 8th Fuzzy Days in Dortmund - Advances in Soft Computing Series, Dortmund, Germany, Springer-Verlag, 2004.

[29] K. Schild, A correspondence theory for terminological logics: preliminary report, Proceedings of IJCAI-91, 12th International Joint Conference on Artificial Intelligence, Sidney, AU, 1991, pp. 466–471.

[30] M. Schmidt-Schaubß, G. Smolka, Attributive concept descriptions with complements, Artificial Intelligence. 48(1) (1991), 1–26.

[31] G. Stoilos, G. Stamou, J.Z. Pan, V. Tzouvaras, I. Horrocks, Reasoning with very expressive fuzzy description logics, Journal of Artificial Intelligence Research. 30(5) (2007) 273–320.

[32] G. Stoilos, U. Straccia, G. Stamou, J.Z. Pan, General concept inclusions in fuzzy description logics, Proceedings of the 17th European Conference on Artificial Intelligence, 2006.

[33] U. Straccia, A fuzzy description logic, Proceedings of the 15th National Conference on Artificial Intelligence, Menlo Park, CA, AAAI Press, 1998, pp. 594–599.

[34] U. Straccia, Reasoning within fuzzy description logics, Journal of Artificial Intelligence Research. 14 (2001) 137–166.

[35] U. Straccia, Transforming fuzzy description logics into classical description logics, Proceedings of the 9th European Conference on Logics in Artificial Intelligence, Lecture Notes in Computer Science, Vol. 3229, Springer-Verlag (2004), pp. 385–399.

[36] U. Straccia, Fuzzy description logic with concrete domains, Tech. Report 2005-TR-03, Istituto di Elaborazione dell'Informazione, January 2005.

[37] U. Straccia, Description logics over lattices, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 14(1) (2006) 1–16.

[38] U. Straccia, F. Bobillo Mixed integer programming, general concept inclusions and fuzzy description logics, Proceedings of the 5th Conference of the European Society for Fuzzy Logic and Technology (2007), pp. 213–220.

[39] C. Tresp, R. Molitor, A description logic for vague knowledge, Proceedings of the 13th European Conference on Artificial Intelligence, Brighton, UK, John Wiley and Sons (1998) pp. 361–365.

[40] W3C, OWL Web Ontology Language Overview, 2004, URL: http://www.w3.org/TR/owl-features/.

[41] L.A. Zadeh, Fuzzy sets, Information and Control. 8 (1965) 338–353.

[42] L.A. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets and Systems. 1(1) (1978) 3–28.