Uncertainty Reasoning for Ontologies with General TBoxes in Description Logic

Volker Haarslev, Hsueh-Ieng Pai, and Nematollaah Shiri

Concordia University
Dept. of Computer Science & Software Engineering
Montreal, Quebec, Canada
{haarslev, hsueh_pa, shiri}@cse.concordia.ca

Abstract. We present a reasoning procedure for ontologies with uncertainty described in Description Logic (DL) which include General TBoxes, i.e., include cycles and General Concept Inclusions (GCIs). For this, we consider the description language $\mathcal{ALC}_U$, in which uncertainty parameters are associated with ABoxes and TBoxes, and which allows General TBoxes. Using this language as a basis, we then present a tableau algorithm which encodes the semantics of the input knowledge base as a set of assertions and linear and/or nonlinear arithmetic constraints on certainty variables. By tuning the uncertainty parameters in the knowledge base, different notions of uncertainty can be modeled and reasoned with, within the same framework. Our reasoning procedure is deterministic, and hence avoids possible empirical intractability in standard DL with General TBoxes. We further illustrate the need for blocking when reasoning with General TBoxes in the context of $\mathcal{ALC}_U$.

1 Introduction

Over the last few years, a number of ontology languages have been developed to help make Web resources more machine-interpretable by giving Web resources a well-defined meaning. Among these languages, the OWL Web Ontology Language [26] 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 standard DLs, such as the one that OWL DL is based on, focus on the classical logic, which is more suitable to describe concepts that are crisp and well-defined in nature. However, in real-world applications, uncertainty is everywhere. In the context of this paper, *uncertainty* refers to a form of deficiency or imperfection in the information for which the truth of such information is not established definitely [15]. The need to model and reason with uncertainty in DLs has been found in many different Semantic Web contexts, such as Semantic Web services, multimedia annotation, and bioinformatics. For example, in an online medical diagnosis system, one might want to express that the certainty of an obese person having a parent

who is obese lies between 0.7 and 1, and John is an obese person with a degree between 0.8 and 1. Such knowledge cannot be expressed nor be reasoned with the standard DLs.

Over the last decade, a number of frameworks have been proposed which extend the standard DLs with uncertainty [5–7, 13, 14, 19–25]. Some of them deal with only vagueness while others deal with only probabilistic knowledge. Since different applications may require to model different notions of uncertainty, and there may be situations in which one needs to model different notions of uncertainty within the same application, we are interested in developing a framework such that different forms of uncertainty knowledge can be represented and reasoned with, in a generic way.

In this paper, we propose a reasoning procedure for uncertainty knowledge bases with General TBoxes by taking a generic approach. As in the standard description languages, a General TBox augments the expressive power of the language by allowing cycles and General Concept Inclusions (GCIs) in the TBox, which in turn allows statements such as domain and range restrictions to be expressed. The proposed reasoning procedure is based on the DL $\mathcal{ALC}_U$, which extends the standard DL $\mathcal{ALC}$ with uncertainty. Inspired by the approach of the parametric framework [16] which incorporates uncertainty in the standard logic programming and deductive databases, the interesting feature of our approach is that, by tuning the uncertainty parameters that are associated with the axioms and assertions in the $\mathcal{ALC}_U$ knowledge bases, different notions of uncertainty can be modeled and reasoned with, using a single reasoning procedure. In addition, since the proposed reasoning procedure is deterministic, it avoids possible empirical intractability in standard DL reasoning when dealing with General TBoxes.

This paper is an extension of our previous work as follows. In [8], we presented a generic framework for representing DLs with uncertainty. That work was further extended in [10] with a core reasoning procedure. A reasoning procedure for dealing with acyclic uncertainty knowledge bases was presented in [9]. In this paper, we further extend [9] by presenting a reasoning procedure that supports uncertainty knowledge bases with General TBoxes.

The rest of this paper is organized as follows. We next review the standard DL $\mathcal{ALC}$ and related work. Section 3 presents the DL $\mathcal{ALC}_U$ and the proposed tableau reasoning algorithm. A detailed example is provided in Section 4 to illustrate the need for blocking when reasoning with $\mathcal{ALC}_U$ knowledge bases which contain General TBoxes. Concluding remarks together with future works are discussed in Section 5.

2 Background and Related Work

In this section, we first review the standard DL language $\mathcal{ALC}$, which is the basis for $\mathcal{ALC}_U$, a generic DL language we proposed which unifies DL frameworks with uncertainty. We will also review related work in this context.

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

Description logics form 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 on which we focus in this paper is called $\mathcal{ALC}$, which corresponds to the propositional multi-modal logic $\mathrm{K}_{(\mathrm{m})}$ [18]. As shown in Fig. 1, the $\mathcal{ALC}$ framework consists of three main components – the description language, the knowledge base, and the reasoning procedure.

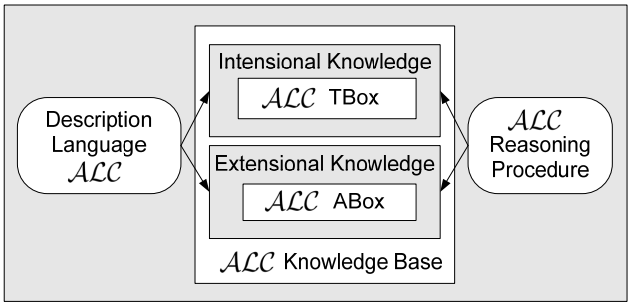

Fig. 1. The $\mathcal{ALC}$ Framework

1. $\mathcal{ALC}$ *Description Language:* Every description language has elementary descriptions which include atomic concepts (unary predicates) and atomic 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 $\top$ as a synonym for $A \sqcup \neg A$, and $\bot$ 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 $a \in \Delta^{\mathcal{I}}$. 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). A General Concept Inclusion (GCI) is a special kind of concept inclusion where the left hand side of the axiom is not restricted to be a concept name but can be an arbitrary concept description. 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 $\langle (a, b) : R \rangle$. 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 reasoning in DL frameworks has been the topic of numerous research for more than a decade [5–7, 13, 14, 19–25]. Some research 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 [12].

Although General TBoxes were supported in [4, 5, 19, 20, 24], there are some major differences between these works and the one we present in this paper. Unlike our reasoning procedure which encode the semantics of the knowledge base as uncertainty constraints, the one proposed in [4] transforms the fuzzy knowledge bases into standard knowledge bases. On the other hand, the reasoning procedure presented in [19, 20] deal with the certainty values directly within the tableau algorithm. Finally, although constraint-based reasoning procedures were also proposed in [5, 24], the main difference between these two and ours is that, while our approach is to develop one reasoning procedure for dealing with uncertainty with different mathematical foundations, these works mainly considered one form. Specifically, [5] supports only product t-norm, and [24] supports only Lukasiewicz semantics.

3 A Reasoning Procedure for $\mathcal{ALC}_U$ Knowledge Bases with General TBoxes

In this section, we present a reasoning procedure for $\mathcal{ALC}_U$ knowledge bases with General TBoxes. For this, we first introduce the DL $\mathcal{ALC}_U$, which extends the standard DL $\mathcal{ALC}$ with uncertainty, by presenting the syntax and semantics of its description language and knowledge base. We then present the proposed $\mathcal{ALC}_U$ reasoning procedure. After that, we illustrate through an example the various extended components of the $\mathcal{ALC}_U$ framework.

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

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.

In order to model different notions of 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 $\mathcal{L}$, and $\otimes$ for

the meet operator (the greatest lower bound). We also assume that there is only one underlying certainty lattice for the entire knowledge base.

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_C, where $CF_C : \Delta^{\mathcal{I}} \rightarrow \mathcal{V}$
- 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 $a \in \Delta^{\mathcal{I}}$

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	$\forall R.C$	$(\forall R.C)^{\mathcal{I}}(a) = \otimes_{b \in \Delta^{\mathcal{I}}}\{f_d(\sim R^{\mathcal{I}}(a,b), C^{\mathcal{I}}(b))\}$

The interpretation of the top concept $\top$ is the greatest element in the certainty domain $\mathcal{V}$, that is, $\top^{\mathcal{I}}(a) = t$, for all $a \in \Delta^{\mathcal{I}}$. For instance, the interpretation of $\top$ is 1 (or true) in the standard logic with $\mathcal{V} = \{0, 1\}$, as well as in other logics with $\mathcal{V} = [0, 1]$. Similarly, the interpretation of the bottom concept $\bot$ is the least element in the certainty domain $\mathcal{V}$, that is, $\bot^{\mathcal{I}}(a) = l$, for all $a \in \Delta^{\mathcal{I}}$. For example, this corresponds to 0 (or false) in the standard logic with $\mathcal{V} = \{0, 1\}$, as well as in other logics with $\mathcal{V} = [0, 1]$.

The semantics of the concept negation $\neg C$ is defined as $(\neg C)^{\mathcal{I}}(a) = \sim C^{\mathcal{I}}(a)$, for all $a \in \Delta^{\mathcal{I}}$. The symbol $\sim$ 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 $\alpha \in \mathcal{V}$.

For example, a common interpretation of $\neg C$ is $1 - C^{\mathcal{I}}(a)$.

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 [16].

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	$\forall \alpha, \beta, \delta \in \mathcal{V}$, $f(\alpha, f(\beta, \delta)) = f(f(\alpha, \beta), \delta)$

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 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 and the algebraic product.

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 and the probability independent functions.

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 here 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.

$\mathcal{ALC}_U$ 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 GCIs 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, 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. In case the choice of the combination function in the current axiom is immaterial, "$-$" is used as a place holder. The concept equation $\langle C \equiv D \mid \alpha, f_c, f_d \rangle$ is equivalent to the two concept inclusions $\langle C \sqsubseteq D \mid \alpha, f_c, f_d \rangle$ and $\langle D \sqsubseteq C \mid \alpha, f_c, f_d \rangle$.

For example, the axiom $\langle \mathit{Rich} \sqsubseteq ((\exists \mathit{owns}.\mathit{ExpensiveCar} \sqcup \exists \mathit{owns}.\mathit{Airplane}) \sqcap \mathit{Golfer}) \mid [0.8, 1], \mathit{min}, \mathit{max} \rangle$ 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*.

In order to transform an axiom of the form $\langle C \sqsubseteq D \mid \alpha, f_c, f_d \rangle$ into its normal form, $\langle \top \sqsubseteq \neg C \sqcup D \mid \alpha, f_c, f_d \rangle$, the semantics of the concept subsumption is restricted to be $f_d(\sim C^{\mathcal{I}}(a), D^{\mathcal{I}}(a))$, 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 $a \in \Delta^{\mathcal{I}}$.

$\mathcal{ALC}_U$ 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 either tall or thin, and smart with probability between 0.6 and 0.8" can be expressed as $\langle \mathit{Mary} : (\mathit{Tall} \sqcup \mathit{Thin}) \sqcap \mathit{Smart} \mid [0.6, 0.8], \times, \mathit{ind} \rangle$. Here, the concept conjunction is interpreted using the algebraic product ($\times$), and the disjunction function is interpreted using the probability independent function (*ind*). Note that, since reasoning with probability often

requires extra information/knowledge about the events and facts in the world, we are investigating ways to model knowledge base with more general probability theory, such as positive/negative correlation, ignorance, and conditional independence.

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

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 procedure "generated individuals"* ($\mathbf{I_G}$)*.*

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* $\mathcal{A}$*.*

Note 1. Since each axiom/assertion in the $\mathcal{ALCC}_U$ knowledge base is associated with a pair of combination functions, different notions of uncertainty (such as fuzzy and simple probability) can be modeled within the same knowledge base, if desired by the user.

3.3 $\mathcal{ALCC}_U$ Reasoning Procedure

Let $\Sigma = \langle \mathcal{T}, \mathcal{A} \rangle$ be an $\mathcal{ALCC}_U$ knowledge base. Fig. 2 gives an overview of the $\mathcal{ALCC}_U$ tableau reasoning procedure. The rectangles represent data or knowledge bases, the arrows show the data flow, and the gray rounded boxes show where data processing is performed.

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

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

Consistency Problem: To check if an $\mathcal{ALCC}_U$ knowledge base $\Sigma = \langle \mathcal{T}, \mathcal{A} \rangle$ is consistent, we first apply the pre-processing steps (see Section 3.3) 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) 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,

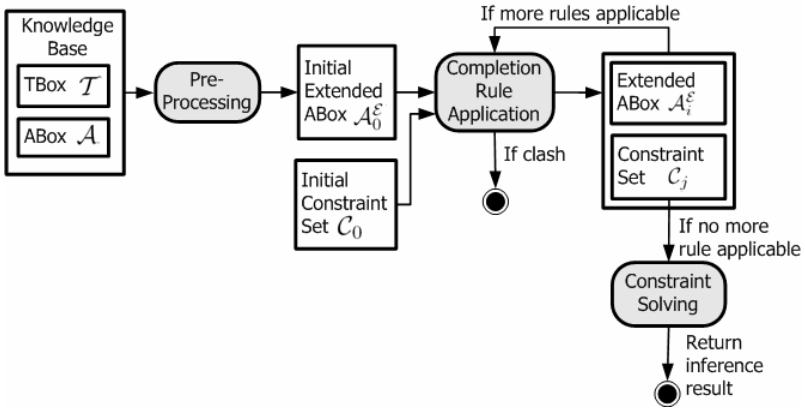

Fig. 2. Reasoning Procedure for $\mathcal{ALC}_U$

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.

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.*

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 the following two axioms: $\langle C \sqsubseteq D \mid \alpha, f_c, f_d\rangle$ and $\langle D \sqsubseteq C \mid \alpha, f_c, f_d\rangle$.
2. Transform every axiom in the TBox into its normal form. That is, replace each axiom of the form $\langle C \sqsubseteq D \mid \alpha, f_c, f_d\rangle$ with $\langle \top \sqsubseteq \neg C \sqcup D \mid \alpha, f_c, f_d\rangle$.
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 $\langle \top \sqsubseteq \neg C \sqcup D \mid \alpha, f_c, f_d\rangle$ in $\mathcal{T}$, we add to $\mathcal{A}$, the assertion $\langle a : \neg C \sqcup D \mid \alpha, f_c, f_d\rangle$.

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

$\mathcal{ALC}_U$ Completion Rules Let $\mathcal{T}$ be a 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 listed in Fig. 3. Since the application of the completion rules may lead to nontermination (see Section 4 for an example), we introduce the notion of blocking to handle this situation.

Definition 4. *(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* $\mathcal{A}_i^{\mathcal{E}}(b) \subseteq \mathcal{A}_i^{\mathcal{E}}(a)$*.*

The purpose of the Clash Triggers is to detect possible inconsistencies in the knowledge base. 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.

The Concept Assertion and Role Assertion rules simply add the certainty value of each atomic concept/role 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 set of constraints $\mathcal{C}_j$.

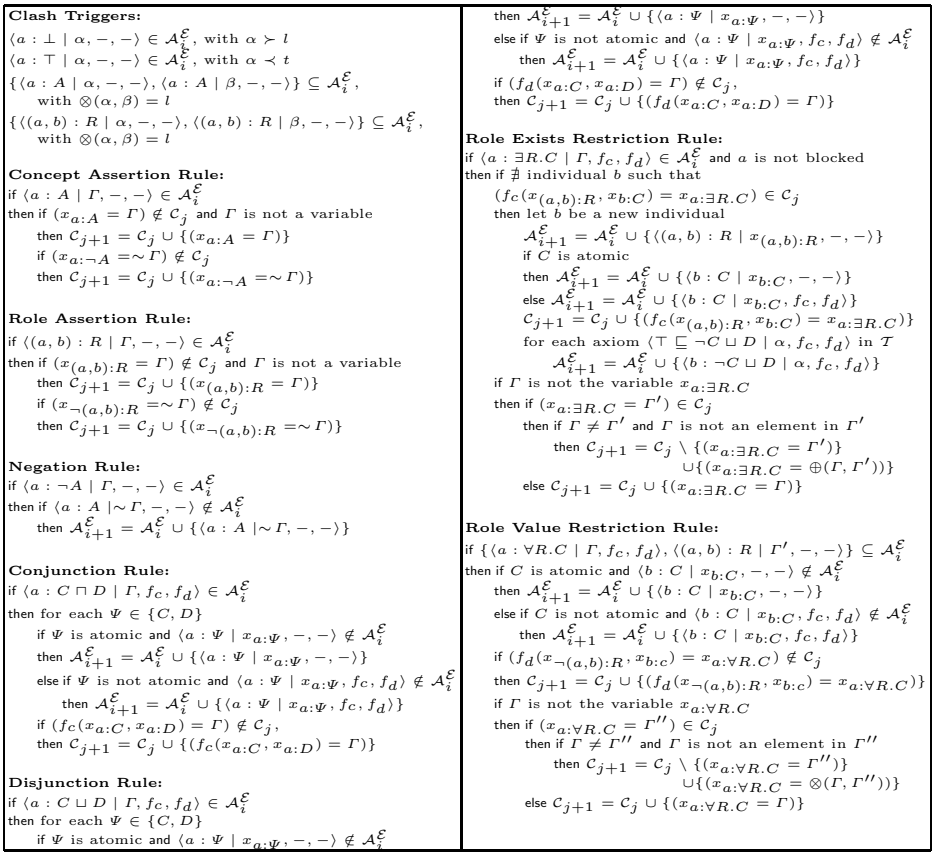

Fig. 3. Completion Rules for $\mathcal{ALC}_U$

The intuition behind the Negation Rule is that, if we know an assertion has a certainty value Γ, then the certainty of its negation can be obtained by applying the negation function to Γ. For example, suppose the certainty domain is $\mathcal{V} = [0, 1]$, and the negation function is defined as $\sim(x) = 1 - x$. If we

have the assertion $\langle John : \neg Tall \mid 0.8, -, - \rangle$ in the ABox, we could also infer $\langle John : Tall \mid 0.2, -, - \rangle$, which is added to the extended ABox.

The Conjunction and Disjunction rules capture the semantics of concept conjunction (resp. disjunction) by applying the conjunction (resp. disjunction) function to the interpretations of $a : C$ and $a : D$. An interesting thing to note is that, in the standard DL, the Disjunction Rule is non-deterministic, since it can be applied in different ways to the same ABox. However, the $\mathcal{ALC}_U$ disjunction rule is deterministic. This is because the semantics of the concept disjunction is 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 infer that $\langle Mary : Tall \mid x_{Mary:Tall}, -, - \rangle$ and $\langle Mary : Thin \mid x_{Mary:Thin}, -, - \rangle$. 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 $x_{Mary:Tall} = x_{Mary:Thin} = 0.8$.

Note 2. Like in standard DLs, our tableau algorithm treats each axiom in $\mathcal{T}$ as a meta constraint. That is, for each individual a in $\mathcal{A}$ and each axiom $\langle C \sqsubseteq D \mid \alpha, f_c, f_d \rangle$ in $\mathcal{T}$, we add $\langle a : \neg C \sqcup D \mid \alpha, f_c, f_d \rangle$ to $\mathcal{A}$. This results in a large number of assertions with concept disjunction be added to $\mathcal{A}$. In standard DLs, this would dramatically extend the search space and is the main cause for empirical intractability, since the disjunction rule in standard DLs is non-deterministic. Therefore, optimization technique like lazy unfolding was introduced in [2]. However, since the disjunction rule in $\mathcal{ALC}_U$ is deterministic, large number of assertions with concept disjunction does not cause problems in our context.

The Role Exists Restriction and Role Value Restriction rules have a similar structure, although they have different semantics. The intuition behind the Role Exists Restriction Rule is that, if we know that an individual a is in $\exists R.C$, there must exist at least one 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, and assert that this individual satisfies all the axioms in the TBox. As usual, the semantics of the Role Exists Restriction is encoded as constraints. On the other hand, 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. The semantics of the Role Value Restriction is also encoded as constraints. For example, suppose the assertion $\langle Tom : \exists hasDisease.Diabetes \mid [0.4, 0.6], min, max \rangle$ is in the extended ABox and the axiom $\langle \top \sqsubseteq \neg Obese \sqcup \exists hasDisease.Diabetes \mid [0.7, 1], \times, ind \rangle$ is in the TBox. Assume also that the ABox originally does not contain any individual b such that *Tom* is related to b through the role *hasDisease*, and b is in the concept *Diabetes*. Then, we could infer $\langle (Tom, d1) : hasDisease \mid x_{(Tom,d1):hasDisease}, -, - \rangle$ and $\langle d1 : Diabetes \mid x_{d1:Diabetes}, -, - \rangle$, where $d1$ is a new individual. In addition, since $d1$ must satisfy the axioms in the TBox, the assertion $\langle d1 : \neg Obese \sqcup \exists hasDisease.Diabetes \mid [0.7, 1], \times, ind \rangle$ is added to the extended ABox. Finally, the constraints

$(min(x_{(Tom,d1):hasDisease}, x_{d1:Diabetes}) = x_{Tom:\exists hasDisease.Diabetes})$ as well as $(x_{Tom:\exists hasDisease.Diabetes} = [0.4, 0.6])$ must be satisfied.

The correctness of the $\mathcal{ALC}_U$ tableau algorithm can be established by showing that it is sound, complete, and it terminates, as shown in [17].

4 An Illustrative Example

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:

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

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 $\langle John : Obese\ Person \mid x_{John:ObesePerson}, -, -\rangle$ 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 $\langle ind1 : ObesePerson \mid x_{ind1:ObesePerson}, -, -\rangle\}$ 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:

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

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:

$$\mathcal{A}_{10}^{\mathcal{E}} = \mathcal{A}_9^{\mathcal{E}} \cup \{\langle (ind1, ind2) : hasParent \mid x_{(ind1,ind2):hasParent}, -, -\rangle\}$$
$$\mathcal{A}_{11}^{\mathcal{E}} = \mathcal{A}_{10}^{\mathcal{E}} \cup \{\langle ind2 : ObesePerson \mid x_{ind2:ObesePerson}, -, -\rangle\}$$
$$\mathcal{C}_9 = \mathcal{C}_8 \cup \{(min(x_{(ind1,ind2):hasParent}, x_{ind2:ObesePerson}) = x_{ind1:\exists hasParent.}$$

$_{ObesePerson})\}$
$\mathcal{A}^{\mathcal{E}}_{12} = \mathcal{A}^{\mathcal{E}}_{11} \cup \{\langle ind2 : (\neg ObesePerson \sqcup \exists hasParent.ObesePerson \mid [0.7, 1], min, max\rangle\}$

Next, the Role Assertion Rule is applied to the assertion in $\mathcal{A}^{\mathcal{E}}_{10}$ 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 $\mathcal{A}^{\mathcal{E}}_{11}$, we obtain:

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

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

$$\mathcal{A}^{\mathcal{E}}_{13} = \mathcal{A}^{\mathcal{E}}_{12} \cup \{\langle ind2 : \neg ObesePerson \mid x_{ind2:\neg ObesePerson}, -, -\rangle\}$$
$$\mathcal{A}^{\mathcal{E}}_{14} = \mathcal{A}^{\mathcal{E}}_{13} \cup \{\langle ind2 : \exists hasParent.ObesePerson \mid x_{ind2:\exists hasParent.ObesePerson}, min, max\rangle\}$$
$$\mathcal{C}_{12} = \mathcal{C}_{11} \cup \{(max(x_{ind2:\neg ObesePerson}, x_{ind2:\exists hasParent.ObesePerson}) = [0.7, 1])\}$$

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.

5 Conclusion and Future Work

In this paper, we presented a tableau reasoning procedure for the DL $\mathcal{ALC}_U$ that is capable of handling General TBoxes. The proposed tableau algorithm derives a set of assertions as well as linear/nonlinear constraints that encode the semantics of the 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 uncertainty parameters that are associated with the axioms and assertions in the knowledge base, different notions of uncertainty can be modeled and reasoned with, using a single reasoning procedure. In addition, the proposed reasoning procedure can handle large number of concept disjunctions caused by the introduction of General TBoxes

without blowing up the search space because our Disjunction Rule is deterministic. We also illustrated through a detailed example the need for blocking in order to ensure termination of the reasoning procedure when General TBoxes are present.

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 [11]. As future work, we plan to extend $\mathcal{ALC}_U$ to support a more expressive portion of DL (e.g., $\mathcal{SHOIN}$, which the popular Web ontology language OWL DL [26] is based on) so that constructors such as number restrictions and transitive properties can be supported. It is also promising to extend the syntax of the description language to support other forms of uncertainty, such as the conditional probability. These additional expressivity would give us more power to handle real-world applications.

Acknowledgement. This work was supported in part by Natural Sciences and Engineering Research Council of Canada, and by ENCS Concordia University.

References

1. Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., and Patel-Schneider, P. F., editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
2. Baader, F., Hollunder, B., Nebel, B., Profitlich, H.-J., and Franconi, E. An empirical analysis of optimization techniques for terminological representation systems or "making KRIS get a move on". In B. Nebel, W. Swartout, and C. Rich, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 3rd International Conference*, pages 270–281, San Mateo, 1992. Morgan Kaufmann.
3. Baader, F., Horrocks, I., and Sattler, U. Description Logics. In Frank van Harmelen, Vladimir Lifschitz, and Bruce Porter, editors, *Handbook of Knowledge Representation*. Elsevier, 2007. To appear.
4. Bobillo, F., Delgado, M., and Gomez-Romero, J. A crisp representation for fuzzy $\mathcal{SHOIN}$ with fuzzy nominals and general concept inclusions. In *Proceedings of the 2nd Workshop on Uncertainty Reasoning for the Semantic Web (URSW)*, Athens, Georgia, USA, November 2006.
5. Bobillo, F. and Straccia, U. A fuzzy description logic with product t-norm. In *Proceedings of the IEEE International Conference on Fuzzy Systems (Fuzz IEEE-07)*, pages 652–657. IEEE Computer Society, 2007.
6. Dürig, M. and Studer, T. Probabilistic ABox reasoning: Preliminary results. In *Proceedings of the International Workshop on Description Logics (DL'05)*, pages 104–111, 2005.
7. Giugno, R. and Lukasiewicz, T. P-$\mathcal{SHOQ}$(D): A probabilistic extension of $\mathcal{SHOQ}$(D) for probabilistic ontologies in the Semantic Web. In *Proceedings of the European Conference on Logics in Artificial Intelligence*, pages 86–97, Cosenza, Italy, 2002. Springer-Verlag. Lecture Notes In Computer Science; Vol. 2424.
8. Haarslev, V., Pai, H.I., and Shiri, N. A generic framework for description logics with uncertainty. In *Proceedings of the 2005 Workshop on Uncertainty Reasoning for the Semantic Web (URSW) at the 4th International Semantic Web Conference*, pages 77–86, Galway, Ireland, November 2005.

9. Haarslev, V., Pai, H.I., and Shiri, N. Completion rules for uncertainty reasoning with the description logic $\mathcal{ALC}$. In *Proceedings of the Canadian Semantic Web Working Symposium - Semantic Web and Beyond: Computing for Human Experience, Vol. 4*, pages 205–225, Quebec City, Canada, June 2006. Springer Verlag.
10. Haarslev, V., Pai, H.I., and Shiri, N. Uncertainty reasoning in description logics: A generic approach. In *Proceedings of the 19th International FLAIRS Conference*, pages 818–823, Melbourne Beach, Florida, May 2006. AAAI Press.
11. Haarslev, V., Pai, H.I., and Shiri, N. Optimizing tableau reasoning in $\mathcal{ALC}$ extended with uncertainty. In *Proceedings of the International Workshop on Description Logics (DL'07)*, pages 307–314, Brixen-Bressanone, Italy, June 2007.
12. Haarslev, V., Pai, H.I., and Shiri, N. Semantic web uncertainty management. In *Encyclopedia of Information Science and Technology, 2nd edition*. Information Science Reference, 2009.
13. Jaeger, M. Probabilistic reasoning in terminological logics. In *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning (KR'94)*, pages 305–316, 1994.
14. Koller, D., Levy, A. Y., and Pfeffer, A. P-CLASSIC: A tractable probablistic description logic. In *Proceedings of the 14th National Conference on Artificial Intelligence*, pages 390–397, Providence, Rhode Island, July 1997. AAAI Press.
15. Lakshmanan, L.V.S. and Shiri, N. Logic programming and deductive databases with uncertainty: A survey. In *Enclyclopedia of Computer Science and Technology*, volume 45, pages 153–176. Marcel Dekker, Inc., New York, 2001.
16. Lakshmanan, L.V.S. and Shiri, N. A parametric approach to deductive databases with uncertainty. *IEEE Transactions on Knowledge and Data Engineering*, 13(4):554–570, 2001.
17. Pai, H.I. *Uncertainty Management for Description Logic-Based Ontologies*. PhD thesis, Concordia University, 2008.
18. Schild, K. A correspondence theory for terminological logics: preliminary report. In *Proceedings of IJCAI-91, 12th International Joint Conference on Artificial Intelligence*, pages 466–471, Sidney, AU, 1991.
19. Stoilos, G., Stamou, G., Pan, J.Z., Tzouvaras, V., and Horrocks, I. Reasoning with very expressive fuzzy description logics. *Journal of Artificial Intelligence Research*, 30:273–320, 2007.
20. Stoilos, G., Straccia, U., Stamou, G., and Pan, J.Z. General concept inclusions in fuzzy description logics. In *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 06)*, 2006.
21. Straccia, U. Reasoning within fuzzy description logics. *Journal of Artificial Intelligence Research*, 14:137–166, 2001.
22. Straccia, U. Uncertainty in description logics: a lattice-based approach. In *Proceedings of the 10th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 251–258, 2004.
23. Straccia, U. Fuzzy description logic with concrete domains. Technical Report 2005-TR-03, Istituto di Elaborazione dell'Informazione, January 2005.
24. Straccia, U. and Bobillo, F. Mixed integer programming, general concept inclusions and fuzzy description logics. In *Proceedings of the 5th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT-07)*, volume 2, pages 213–220, Ostrava, Czech Republic, 2007. University of Ostrava.
25. Tresp, C. and Molitor, R. A description logic for vague knowledge. In *Proceedings of ECAI-98*, pages 361–365, Brighton, UK, 1998. John Wiley and Sons.
26. W3C. OWL web ontology language overview, 2004.
URL: http://www.w3.org/TR/owl-features/.