J Philos Logic manuscript No.
(will be inserted by the editor)

On the Decidability Status of Fuzzy $\mathcal{ALC}$ with General Concept Inclusions

Franz Baader · Stefan Borgwardt · Rafael Peñaloza

Received: date / Accepted: date

Abstract The combination of Fuzzy Logics and Description Logics (DLs) has been investigated for at least two decades because such fuzzy DLs can be used to formalize imprecise concepts. In particular, tableau algorithms for crisp Description Logics have been extended to reason also with their fuzzy counterparts. It has turned out, however, that in the presence of general concept inclusion axioms (GCIs) this extension is less straightforward than thought. In fact, a number of tableau algorithms claimed to deal correctly with fuzzy DLs with GCIs have recently been shown to be incorrect. In this paper, we concentrate on fuzzy $\mathcal{ALC}$, the fuzzy extension of the well-known DL $\mathcal{ALC}$. We present a terminating, sound, and complete tableau algorithm for fuzzy $\mathcal{ALC}$ with arbitrary continuous t-norms. Unfortunately, in the presence of GCIs, this algorithm does not yield a decision procedure for consistency of fuzzy $\mathcal{ALC}$ ontologies since it uses as a sub-procedure a solvability test for a finitely represented, but possibly infinite, system of inequations over the real interval $[0, 1]$, which are built using the t-norm. In general, it is not clear whether this solvability problem is decidable for such infinite systems of inequations. This may depend on the specific t-norm used. In fact, we also show in this paper that consistency of fuzzy $\mathcal{ALC}$ ontologies with GCIs is undecidable for the product t-norm. This implies, of course, that for the infinite systems of inequations produced by the tableau algorithm for fuzzy $\mathcal{ALC}$ with product t-norm, solvability is in general undecidable. We also give a brief overview of recently obtained (un)decidability results for fuzzy $\mathcal{ALC}$ w.r.t. other t-norms.

Keywords Fuzzy Description Logics · Decidability

Partially supported by the Deutsche Forschungsgemeinschaft (DFG) under grant BA 1122/17-1 and within the Cluster of Excellence 'Center for Advancing Electronics Dresden'.

F. Baader · S. Borgwardt · R. Peñaloza
Institute of Theoretical Computer Science, Technische Universität Dresden, 01062, Dresden, Germany
E-mail: {baader,stefborg,penaloza}@tcs.inf.tu-dresden.de

F. Baader · R. Peñaloza
Center for Advancing Electronics Dresden

1 Introduction

Description logics (DLs) [2] are a family of logic-based knowledge representation formalisms, which can be used to represent the conceptual knowledge of an application domain in a structured and formally well-understood way. They are employed in various application domains, such as natural language processing, configuration, and databases, but their main breakthrough arguably came with the adoption of the DL-based language OWL [19] as standard ontology language for the semantic web. Another successful application area for DLs is the definition of medical ontologies, such as SNOMED CT[1] and GALEN.[2]

In Description Logics, concepts are formally described by *concept descriptions*, i.e., expressions that are built from concept names (unary predicates) and role names (binary predicates) using concept constructors. The expressivity of a particular DL is determined by the concept constructors available in it. From a semantic point of view, concept names and concept descriptions represent sets of individuals, whereas roles represent binary relations between individuals. For example, using the concept names Patient and Running-nose, and the role name hasSymptom, the concept of all *patients with running noses* can be represented by the concept description

$$\mathsf{Patient} \sqcap \exists \mathsf{hasSymptom}.\mathsf{Running\text{-}nose}.$$

In addition to the description language (i.e., the formalism for constructing concept descriptions), DLs provide their users with a terminological and an assertional formalism. In its simplest form, a DL *terminology* (usually called *TBox*) can be used to introduce abbreviations for complex concept descriptions. For example, the *concept definition*

$$\mathsf{Private\text{-}patient} \equiv \mathsf{Patient} \sqcap \exists \mathsf{hasInsurance}.\mathsf{Private\text{-}health}$$

expresses that private patients are patients that have a private health insurance. So-called *general concept inclusions (GCIs)* can be used to state additional constraints on the interpretation of concepts and roles. In our medical example, one could express that patients with running noses have a cold or hay fever using the GCI

$$\mathsf{Patient} \sqcap \exists \mathsf{hasSymptom}.\mathsf{Running\text{-}nose} \sqsubseteq \exists \mathsf{hasDisease}.(\mathsf{Cold} \sqcup \mathsf{Hay\text{-}fever}).$$

Note that the concept definition $A \equiv C$ can be expressed using the GCIs $A \sqsubseteq C$ and $C \sqsubseteq A$.

In the *assertional part (ABox)* of a DL-based ontology, facts about a specific application situation can be stated by introducing named individuals and relating them to concepts and roles. For example, the assertions

$$\mathsf{Patient}(\mathsf{linda}), \quad \mathsf{Private\text{-}health}(\mathsf{AXA\text{-}PPP}), \quad \mathsf{hasInsurance}(\mathsf{linda}, \mathsf{AXA\text{-}PPP}),$$

state that Linda is a patient that has the private health insurance AXA-PPP. An *ontology* is a TBox together with an ABox, i.e., finite set of GCIs and assertions.

Knowledge representation systems based on DLs provide their users with various inference services that allow them to deduce implicit consequences from the explicitly represented knowledge. For example, given the concept definition and the assertions introduced above, one can deduce the assertion $\mathsf{Private\text{-}patient}(\mathsf{linda})$, i.e., that Linda is a private patient. An important inference service for DL-based ontologies is testing their consistency, i.e., checking whether a given ontology is non-contradictory by testing whether it has a model. Indeed,

[1] http://www.ihtsdo.org/snomed-ct/

[2] http://www.opengalen.org/

for crisp DLs that are closed under all Boolean operations, all the other standard inference problems can be reduced to consistency. For such DLs, tableau algorithms [5] are still the method of choice to obtain practical inference procedures. In principle, to decide consistency of a given ontology, such an algorithm tries to generate a finite model for this ontology by decomposing complex concept assertions according to the semantics of the concept constructors. For example, to satisfy the assertion

$$(\exists \mathsf{hasInsurance}.\mathsf{Private\text{-}health})(\mathsf{linda}),$$

a tableau algorithm would introduce a new individual name, say INS, and generate the assertions

$$\mathsf{hasInsurance}(\mathsf{linda},\mathsf{INS}),\ \ \mathsf{Private\text{-}health}(\mathsf{INS}).$$

In the presence of GCIs, introducing such new individuals may cause a non-terminating execution. For example, assume that the ontology contains the GCI $\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\mathsf{Human}$ and the assertion $\mathsf{Human}(\mathsf{linda})$. The tableau algorithm would first add the assertion

$$(\exists \mathsf{hasParent}.\mathsf{Human})(\mathsf{linda}),$$

and then a new individual name, say CHI_1, together with the assertions

$$\mathsf{hasParent}(\mathsf{linda},\mathsf{CHI}_1),\ \ \mathsf{Human}(\mathsf{CHI}_1).$$

But now CHI_1 is a human being, and thus needs a human parent CHI_2, which again needs a human parent, and so on. Thus, without additional precautions, the tableau algorithm would not terminate. In the DL community, these precautions are called *blocking*. Basically, blocking can be used to detect that the same situation that leads to the creation of new individuals occurs repeatedly, and then blocks the application of the tableau rule that creates the new individual (see [5] for details). This way, termination of the tableau algorithm can be regained for the DL $\mathcal{ALC}$ considered in this paper, and also for various more expressive DLs (see, e.g., [20]). However, for some DLs adding GCIs actually makes the consistency problem undecidable [1, 23].

Fuzzy variants of Description Logics (DLs) were introduced in order to deal with applications where membership to concepts cannot always be determined in a precise way. For example, assume that we want to express that a patient that has a high temperature and a running nose has a cold using the GCI

$$\mathsf{Patient} \sqcap \exists \mathsf{hasSymptom}.\mathsf{Running\text{-}nose} \sqcap \exists \mathsf{hasTemperature}.\mathsf{High} \sqsubseteq \exists \mathsf{hasDisease}.\mathsf{Cold}.$$

Here it makes sense to view High as a fuzzy concept, to which 36°C belongs with a low membership degree (say 0.2), 38°C with a higher membership degree (say 0.7), and 40°C with an even higher membership degree (say 0.9). In the presence of such fuzzy concepts, ABox assertions must then be equipped with a membership degree. For example, the assertion $\langle \mathsf{High}(\mathsf{T}_1) \geq 0.8 \rangle$ says that temperature T_1 is high with membership degree at least 0.8. If we are not so sure about the measurement (e.g., if it was taken under the armpit), we could also equip the role assertion $\mathsf{hasTemperature}(\mathsf{linda},\mathsf{T}_1)$ with a membership degree smaller than 1. The use of fuzzy concepts in medical applications is, for instance, described in more detail in [25].

A great variety of fuzzy DLs have been investigated in the literature [22, 16]. In fact, compared to crisp DLs, fuzzy DLs offer an additional degree of freedom when defining their expressiveness: in addition to deciding which concept constructors (like conjunction $\sqcap$, disjunction $\sqcup$, existential restriction $\exists r.C$) and which terminological formalism (like no TBox,

acyclic concept definitions, general concept inclusions) to use, one must also decide how to interpret the concept constructors by appropriate functions on the domain of fuzzy values $[0,1]$. For example, conjunction can be interpreted by different t-norms $\otimes$ (such as Gödel, Łukasiewicz, and product) [21] and there are also different options for how to interpret negation (such as involutive negation and residual negation). In addition, one can either consider all models or only so-called witnessed models [18] when defining the semantics of fuzzy DLs. Here, we will restrict the attention to witnessed models.

Decidability of fuzzy DLs is often shown by adapting the tableau algorithms for the corresponding crisp DL to the fuzzy case. This was first done for the case of DLs without GCIs [33,31,29,9], but then also extended to GCIs [30,32,7,8]. Usually, these tableau algorithms reason w.r.t. witnessed models.[3] In principle, the extended tableau algorithms generate a system of inequations that constrain the possible fuzzy degrees. For example, if the assertion $(A \sqcap B)(a)$ is supposed to hold with degree at least 0.7, then $A(a)$ needs to hold with a degree d_1 and $B(a)$ with a degree d_2 such that $d_1 \otimes d_2 \geq 0.7$. After termination of the tableau algorithm, one then needs to check the system of inequations for solvability. If the algorithm is sound and complete, then the input ontology is consistent iff the system of inequations produced by the tableau algorithm is solvable. As mentioned before, most of the tableau algorithms for fuzzy DLs that were claimed to deal with GCIs are actually not correct. The reason for this is that they use a blocking approach that does not take the system of inequations into account in an appropriate way. In fact, some of fuzzy DLs in question have later been shown to be undecidable [3,4,13,15].

The goal of this paper is to elucidate the causes of (un)decidability in fuzzy DLs with GCIs. To achieve this, we proceed in the following two directions. First, we introduce a more sophisticated blocking condition for the tableau algorithms that takes into account also the system of inequations produced by the tableau. We show that the use of this blocking condition ensures termination, and that the obtained tableau algorithm is sound and complete in the following sense: after termination, the computed system of inequations together with the blocking information uniquely determines a possibly infinite system of inequations such that the input ontology is consistent iff this system is solvable.

Though solvability of finite systems of inequations is usually decidable, it is not clear how to decide solvability of such finitely represented infinite systems. Thus, despite our tableau algorithm being terminating, as well as sound and complete, it does not constitute a decision procedure since it requires the solution of a problem whose decidability status is not clear. Thus, undecidability is not caused by the impossibility of finding appropriate blocking conditions, but rather by the inability of deciding whether the generated system of inequations is contradictory or not.

Second, we examine the particular example of fuzzy $\mathcal{ALC}$ with the product t-norm to demonstrate the basic ideas underlying the recent undecidability proofs for fuzzy DLs in the presence of GCIs. This is done by a reduction from the Post Correspondence Problem, which is well-known to be undecidable.

In brief, the main goal of this paper is to provide an intuitive understanding of what makes reasoning in fuzzy DLs undecidable. To achieve this, we

- provide a sound, complete and terminating algorithm for fuzzy $\mathcal{ALC}$;
- show why this algorithm is not a decision procedure for this logic; and
- provide a prototypical proof of undecidability.

[3] In fact, witnessed models were introduced in [18] to correct the proof of correctness for the tableau algorithm presented in [33].

In the next section, we will introduce the (crisp) DL $\mathcal{ALC}$ and sketch a tableau algorithm for this DL. In Section 3 we introduce t-norms and fuzzy logic, and in the subsequent section we define fuzzy $\mathcal{ALC}$. In Section 5, we introduce a tableau algorithm for fuzzy $\mathcal{ALC}$, first without and then with GCIs, and prove that it is terminating as well as sound and complete in the sense introduced above. This algorithm is parameterized on the employed t-norm. In Section 6, we show that GCIs actually cause undecidability of consistency of fuzzy $\mathcal{ALC}$ with product t-norm. In addition, we review some of the recently obtained (un)decidability results for other t-norms.

2 The Description Logic $\mathcal{ALC}$

Description logics (DLs) [2] are logic-based knowledge representation formalisms tailored towards representing the conceptual knowledge of an application domain in a structured and well-understood way. In these logics, knowledge is expressed through *concept descriptions* that are built from atomic concepts (corresponding to unary predicates from first-order logic) and atomic roles (binary predicates) using a set of constructors, such as conjunction ($\sqcap$) or existential restrictions ($\exists$). We focus on the description logic $\mathcal{ALC}$, the smallest DL that is closed under propositional constructors and allows existential and value restrictions.

Definition 2.1 Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be two disjoint sets of *concept names* and *role names*, respectively. The set of $\mathcal{ALC}$ *concept descriptions* is the smallest set containing $\mathsf{N_C}$ such that:

- if C and D are $\mathcal{ALC}$ concept descriptions, then so are $\neg C, C \sqcap D$, and $C \sqcup D$; and
- if C is an $\mathcal{ALC}$ concept description and $r \in \mathsf{N_R}$, then $\exists r.C$ and $\forall r.C$ are $\mathcal{ALC}$ concept descriptions.

For example, using the concept names Human and Male, and the role name $\mathsf{hasParent}$, the concept $\mathsf{Male} \sqcap \exists \mathsf{hasParent}.\mathsf{Human}$ expresses all male individuals that have a human parent.

Concept names can be seen as unary predicates of first-order logic, while role names correspond to binary predicates. The semantics of concept descriptions is consequently defined using interpretations that assign sets to concept descriptions and binary relations to roles.

Definition 2.2 An *interpretation* is a tuple of the form $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty set called the *domain* and $\cdot^{\mathcal{I}}$ is a function that assigns to every concept name A a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and to every role name r a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.

This function is extended to $\mathcal{ALC}$ concept descriptions as follows:

- $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$;
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$;
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$;
- $(\exists r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \text{there is } (x,y) \in r^{\mathcal{I}} \text{ with } y \in C^{\mathcal{I}}\}$;
- $(\forall r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \text{for all } (x,y) \in r^{\mathcal{I}} \text{ it holds that } y \in C^{\mathcal{I}}\}$.

Similar to first-order logic, existential and value restrictions are dual to each other, i.e., the interpretations of the concepts $\exists r.\neg C$ and $\neg \forall r.C$ are always the same.

In description logics, knowledge is represented using a set of assertional axioms expressing properties of specific individuals of the domain, and terminological axioms, which restrict the interpretations of concept descriptions. For example, the assertion $\mathsf{Male}(\mathsf{chiron})$ and

Table 2.1 Tableau rules of the consistency algorithm for $\mathcal{ALC}$.

($\sqcap$) if $(C \sqcap D)(x) \in \mathcal{A}$ but $\{C(x), D(x)\} \not\subseteq \mathcal{A}$, then $\mathcal{A}' := \mathcal{A} \cup \{C(x), D(x)\}$

($\sqcup$) if $(C \sqcup D)(x) \in \mathcal{A}$ but $\{C(x), D(x)\} \cap \mathcal{A} = \emptyset$ then $\mathcal{A}' := \mathcal{A} \cup \{C(x)\}$, and $\mathcal{A}'' := \mathcal{A} \cup \{D(x)\}$

($\exists$) if $(\exists r.C)(x) \in \mathcal{A}$ but there is no z such that $\{r(x,z), C(z)\} \subseteq \mathcal{A}$ then $\mathcal{A}' := \mathcal{A} \cup \{r(x,y), C(y)\}$, where y is an individual name not occurring in $\mathcal{A}$

($\forall$) if $\{(\forall r.C)(x), r(x,y)\} \subseteq \mathcal{A}$ but $C(y) \notin \mathcal{A}$, then $\mathcal{A}' := \mathcal{A} \cup \{C(y)\}$

the terminological axiom $\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\mathsf{Human}$ express that the named individual Chiron is male, and that every human has a human parent, respectively.

To correctly deal with the assertional knowledge, we additionally consider a set $\mathsf{N_I}$ of *individual names* that is disjoint with $\mathsf{N_C}$ and $\mathsf{N_R}$. The interpretation $\mathcal{I}$ maps every individual name $a \in \mathsf{N_I}$ to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$.

Definition 2.3 Let C be a concept description, r a role name, and $a, b \in \mathsf{N_I}$. An *assertion* is of the form $C(a)$ (called *concept assertion*) or $r(a,b)$ (called *role assertion*). An *ABox* is a finite set of assertions.

A *general concept inclusion* (GCI) is of the form $C \sqsubseteq D$, where C, D are concept descriptions. A finite set of GCIs is called a *TBox*.

An interpretation $\mathcal{I}$ *satisfies* the concept assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, *satisfies* the role assertion $r(a,b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$, and *satisfies* the GCI $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. This interpretation $\mathcal{I}$ is a *model* of the ABox $\mathcal{A}$ if it satisfies all assertions in $\mathcal{A}$, and is a *model* of the TBox $\mathcal{T}$ if it satisfies all GCIs in $\mathcal{T}$.

An ABox $\mathcal{A}$ is *consistent* w.r.t. a TBox $\mathcal{T}$ if there exists an interpretation that is a model of both $\mathcal{A}$ and $\mathcal{T}$. We say that a concept description C is *subsumed* by a concept description D w.r.t. a TBox $\mathcal{T}$ if every model of $\mathcal{T}$ satisfies the GCI $C \sqsubseteq D$.

The central reasoning task in DLs is to decide consistency of an ABox w.r.t. a TBox. Many other reasoning problems, such as subsumption, can be reduced to the consistency problem. In fact, C is subsumed by D w.r.t. $\mathcal{T}$ iff the singleton ABox $\{C \sqcap \neg D(a)\}$ is inconsistent w.r.t. $\mathcal{T}$ for an arbitrary individual name a. For the rest of this paper, we focus only on deciding consistency.

The most widely used reasoning technique for DLs is the tableau-based approach, which was first introduced in the context of DLs in [28]. We now describe this technique for deciding consistency of an ABox assuming an empty TBox, and later show how to extend it to work in the presence of GCIs.

2.1 Consistency without TBoxes

The tableau-based approach for deciding consistency of an ABox is based on the fact that $\mathcal{ALC}$ has the *finite model property*, i.e., every consistent ABox has a *finite* model. Given an ABox $\mathcal{A}_0$, the tableau algorithm for consistency tries to construct a finite interpretation $\mathcal{I}$ that is a model of $\mathcal{A}_0$. We assume that every concept description is in *negation normal form* (NNF), where negation appears only directly in front of concept names. Every concept description can be transformed to NNF in linear time using de Morgan's rules, the duality of quantifiers, and elimination of double negations.

The algorithm starts with $\mathcal{A}_0$ and applies the consistency-preserving rules shown in Table 2.1 to it. The transformation rule ($\sqcup$) is *non-deterministic* in the sense that a given ABox

$\mathcal{A}$ is transformed into two new ABoxes such that $\mathcal{A}$ is consistent iff *one* of the new ABoxes is also consistent. For this reason, we consider finite sets of ABoxes $\mathcal{S} = \{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$ instead of a single ABox. Such a set is *consistent* iff there is some i, $1 \leq i \leq k$, where $\mathcal{A}_i$ is consistent. A rule from Table 2.1 is applied to a given finite set of ABoxes $\mathcal{S}$ as follows: it takes an element $\mathcal{A}$ of $\mathcal{S}$ and replaces it by one ABox $\mathcal{A}'$ or by two ABoxes $\mathcal{A}', \mathcal{A}''$.

Definition 2.4 The ABox $\mathcal{A}$ is *complete* iff none of the transformation rules from Table 2.1 applies to it. It contains a *clash* if $\{A(x), \neg A(x)\} \subseteq \mathcal{A}$ for some concept name A and individual name x. It is *closed* if it contains a clash, and *open* otherwise.

The *consistency algorithm* for $\mathcal{ALC}$ works as follows. It starts with the singleton set of ABoxes $\{\mathcal{A}_0\}$ and applies the rules from Table 2.1 in arbitrary order until no more rules apply. It answers "consistent" if the set $\mathcal{S}$ of ABoxes obtained this way contains an open ABox, and "inconsistent" otherwise. The fact that this algorithm is a decision procedure for consistency of $\mathcal{ALC}$ ABoxes is an easy consequence of the following facts [5]:

1. there is no infinite sequence of rule applications starting with $\{\mathcal{A}_0\}$;
2. the transformation rules preserve consistency; that is, if $\mathcal{S}'$ is obtained from $\mathcal{S}$ by the application of a transformation rule, then $\mathcal{S}$ is consistent iff $\mathcal{S}'$ is consistent;
3. any complete and open ABox is consistent; and
4. any closed ABox is inconsistent.

We now provide more details on the reasons for termination of the algorithm. The transformation rules are monotonic in the sense that every application of a rule to $\mathcal{A}$ adds a new concept assertion and does not remove anything. Additionally, all concept descriptions appearing in $\mathcal{A}$ are subconcepts of concept descriptions occurring in the initial ABox $\mathcal{A}_0$. Together, these two facts imply that there can only be a finite number of rule application per individual. Moreover, only the existential rule can introduce new individuals. Every existentially quantified assertion in $\mathcal{A}$ can trigger at most one introduction of a new individual, and hence the number of successors of an individual in $\mathcal{A}$ is bounded by the number of existential restrictions in $\mathcal{A}_0$. The length of successor chains of new individuals in $\mathcal{A}$ is also bounded by the maximal size of the concept descriptions occurring in $\mathcal{A}$. Thus, after a finite amount of rule applications, all the ABoxes are complete.

2.2 Consistency w.r.t. TBoxes

If the TBox $\mathcal{T}$ is not empty, then every individual of the interpretation generated by the tableau algorithm must also satisfy the restrictions imposed by the GCIs in $\mathcal{T}$. If $C \sqsubseteq D \in \mathcal{T}$, then every individual must belong to the concept $\mathsf{nnf}(\neg C \sqcup D)$.[4] To impose this restriction, it suffices to include the new rule ($\sqsubseteq$):

($\sqsubseteq$) if we have $C \sqsubseteq D \in \mathcal{T}$, but $(\mathsf{nnf}(\neg C \sqcup D))(x) \notin \mathcal{A}$ for some $x \in \mathsf{N_I}$ occurring in $\mathcal{A}$, then set $\mathcal{A}' := \mathcal{A} \cup \{(\mathsf{nnf}(\neg C \sqcup D))(x)\}$.

It is easy to see that this algorithm is sound and complete for ABox consistency w.r.t. TBoxes. However, it may not terminate, as shown by the following example.

Example 2.5 Consider the ABox $\mathcal{A}_0 := \{\mathsf{Human}(x_0)\}$ and the TBox

$$\{\mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\mathsf{Human}\}.$$

[4] $\mathsf{nnf}(C)$ denotes the negation normal form of the concept C.

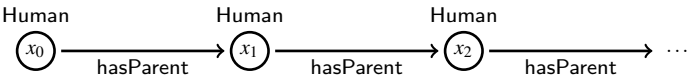

Fig. 2.1 An infinite ABox.

An application of the ($\sqsubseteq$) rule adds the assertion $\neg\mathsf{Human} \sqcup \exists\mathsf{hasParent}.\mathsf{Human}(x_0)$ to $\mathcal{A}_0$. Applying the ($\sqcup$) and ($\exists$) rules yields a new ABox $\mathcal{A}_1$ containing the two assertions $\mathsf{hasParent}(x_0,x_1)$ and $\mathsf{Human}(x_1)$. By repeating this argument, we generate an infinite sequence of sets of ABoxes $\mathcal{S}_0,\mathcal{S}_1,\ldots$ and individuals $x_1,x_2,\ldots$ such that the ABox

$$\mathcal{A}_{i+1} \supseteq \mathcal{A}_i \cup \{\mathsf{hasParent}(x_i,x_{i+1}), \mathsf{Human}(x_{i+1}), \exists\mathsf{hasParent}.\mathsf{Human}(x_{i+1})\}$$

is in $\mathcal{S}_{i+1}$ for every $i \geq 0$ (see Figure 2.1). Notice that every individual x_i, $i \geq 1$, receives the same concept assertions as x_1; intuitively, we can say that the procedure has run into a cycle.

To regain termination, one can use a cycle-detection mechanism to restrict the application of rules generating new individuals—here, the ($\exists$) rule. Let $\mathcal{L}_\mathcal{A}(x) := \{C \mid C(x) \in \mathcal{A}\}$ denote the concept descriptions associated to the individual x in an ABox $\mathcal{A}$. We say that the application of the ($\exists$) rule to an individual x is *blocked* by an individual y in an ABox $\mathcal{A}$ if $\mathcal{L}_\mathcal{A}(x) = \mathcal{L}_\mathcal{A}(y)$.[5] The main idea behind blocking is that an open and complete ABox describes a cyclic model, in which the blocked individual x reuses the role successors of its blocking node y instead of generating new ones. For instance, rather than generating a new hasParent-successor for x_2 in Example 2.5, one can simply reuse the hasParent-successor of x_1.

To avoid a situation of cyclic blocking where x and y mutually block each other, one can consider an enumeration of all individual names, for instance in the order in which they are generated by rule applications, and require that nodes can only be blocked by other nodes appearing earlier in the enumeration. This blocking condition does not affect the soundness and completeness of the tableau procedure, but suffices to regain termination [5, 14].

3 Triangular Norms and Fuzzy Logic

Fuzzy logics are formalisms introduced to express imprecise or vague information [34, 17]. They extend classical logic by interpreting predicates as fuzzy sets, rather than crisp sets, over an interpretation domain.

Given a non-empty domain Δ, a *fuzzy set* is a function $F\colon \Delta \to [0,1]$ from Δ into the real unit interval $[0,1]$, expressing that an element $x \in \Delta$ belongs to F with *degree* $F(x)$. The interpretation of the logical constructors is based on appropriate truth functions that generalize the properties of the connectives of classical logic to the interval $[0,1]$. The most prominent truth functions used in the fuzzy logic literature are based on triangular norms (t-norms) [21]. A *t-norm* is an associative, commutative binary operator $\otimes\colon [0,1]\times[0,1] \to [0,1]$ that has unit 1, and is monotonic, i.e., for every $\xi,\chi,\zeta \in [0,1]$, if $\xi \leq \chi$, then $\xi \otimes \zeta \leq \chi \otimes \zeta$. A t-norm is called *continuous* if it is continuous as a function from $[0,1]\times[0,1]$ to $[0,1]$. Here we consider only continuous t-norms and often call them simply t-norms.

[5] For the case of $\mathcal{ALC}$, the weaker blocking condition $\mathcal{L}_\mathcal{A}(x) \subseteq \mathcal{L}_\mathcal{A}(y)$ would suffice. We describe equality blocking, as it is closer to the blocking condition introduced later for fuzzy DLs (see Definition 5.8).

Table 3.1 The three fundamental continuous t-norms.

Name	t-norm ($\xi \otimes \chi$)	t-conorm ($\xi \oplus \chi$)	residuum ($\xi \Rightarrow \chi$)
Gödel (G)	$\min\{\xi, \chi\}$	$\max\{\xi, \chi\}$	$\begin{cases} 1 & \text{if } \xi \leq \chi \\ \chi & \text{otherwise} \end{cases}$
product (Π)	$\xi \cdot \chi$	$\xi + \chi - \xi \cdot \chi$	$\begin{cases} 1 & \text{if } \xi \leq \chi \\ \chi/\xi & \text{otherwise} \end{cases}$
Łukasiewicz (Ł)	$\max\{\xi + \chi - 1, 0\}$	$\min\{\xi + \chi, 1\}$	$\min\{1 - \xi + \chi, 1\}$

Every continuous t-norm $\otimes$ has a unique *residuum* $\Rightarrow$, which is defined as

$$\xi \Rightarrow \chi := \sup\{\zeta \in [0,1] \mid \xi \otimes \zeta \leq \chi\}.$$

Based on this residuum, the unary *residual negation* is defined as $\ominus\xi = \xi \Rightarrow 0$.[6] To generalize disjunction, the *t-conorm* $\oplus$, given by $\xi \oplus \chi = 1 - ((1-\xi) \otimes (1-\chi))$, is used. Three important continuous t-norms with their t-conorms and residua are depicted in Table 3.1. These are *fundamental* in the sense that every continuous t-norm can be constructed from these three as follows.

Definition 3.1 Let I be a set and for each $i \in I$ let $\otimes_i$ be a continuous t-norm and $\alpha_i, \beta_i \in [0,1]$ such that $\alpha_i < \beta_i$ and the intervals (α_i, β_i) are pairwise disjoint. The *ordinal sum* of the t-norms $\otimes_i$ is the t-norm $\otimes$ with

$$\xi \otimes \chi = \begin{cases} \alpha_i + (\beta_i - \alpha_i)\left(\frac{\xi-\alpha_i}{\beta_i-\alpha_i} \otimes_i \frac{\chi-\alpha_i}{\beta_i-\alpha_i}\right) & \text{if } \xi, \chi \in [\alpha_i, \beta_i],\ i \in I, \\ \min\{\xi, \chi\} & \text{otherwise.} \end{cases}$$

The ordinal sum of a class of continuous t-norms is itself a continuous t-norm, and its residuum is given by

$$\xi \Rightarrow \chi = \begin{cases} 1 & \text{if } \xi \leq \chi, \\ \alpha_i + (\beta_i - \alpha_i)\left(\frac{\xi-\alpha_i}{\beta_i-\alpha_i} \Rightarrow_i \frac{\chi-\alpha_i}{\beta_i-\alpha_i}\right) & \text{if } \alpha_i \leq \chi < \xi \leq \beta_i,\ i \in I, \\ \chi & \text{otherwise,} \end{cases}$$

where $\Rightarrow_i$ is the residuum of $\otimes_i$, for each $i \in I$. Intuitively, this means that at each interval $[\alpha_i, \beta_i]$, a scaled-down and repositioned copy of the t-norm $\otimes_i$ and its residuum $\Rightarrow_i$ is used. For elements not belonging to the same interval, the Gödel t-norm is used.

Theorem 3.2 ([26]) *Every continuous t-norm is isomorphic to the ordinal sum of copies of the Łukasiewicz and product t-norms.*

Motivated by this representation as an ordinal sum, we say that a continuous t-norm $\otimes$ *starts with the Łukasiewicz t-norm* if in its representation as ordinal sum there is an $i \in I$ such that $\alpha_i = 0$ and $\otimes_i$ is isomorphic to the Łukasiewicz t-norm.

An additional property that will be useful for characterizing decidability of fuzzy DLs is the presence of zero divisors. An element $\xi \in (0,1)$ is called a *zero divisor* for $\otimes$ if there is a $\zeta \in (0,1)$ such that $\xi \otimes \zeta = 0$. Of the three fundamental continuous t-norms, only the Łukasiewicz t-norm has zero divisors. In fact, every element ξ in the interval $(0,1)$ is a zero divisor for this t-norm since $(1-\xi) \in (0,1)$ and $\xi \otimes_{Ł} (1-\xi) = \max\{\xi + 1 - \xi - 1, 0\} = 0$. Moreover, a continuous t-norm can only have zero divisors if it starts with the Łukasiewicz t-norm.

[6] The residual negation is also called *precomplement*.

Lemma 3.3 ([21]) *A continuous t-norm has zero divisors iff it starts with the Łukasiewicz t-norm.*

4 Fuzzy Description Logics

Just as classical description logics, fuzzy DLs are based on concept descriptions built from the mutually disjoint sets $\mathsf{N_C}, \mathsf{N_R}$ and $\mathsf{N_I}$ of *concept names*, *role names*, and *individual names*, respectively, using concept constructors. The syntax of the fuzzy variant of $\mathcal{ALC}$ is exactly the same as that of classical $\mathcal{ALC}$ (see Definition 2.1). Compared to classical DLs, fuzzy DLs have an additional degree of freedom in the selection of their semantics since the interpretation of the constructors depends on the specific t-norm chosen. Given a continuous t-norm $\otimes$, we obtain the fuzzy DL $\otimes$-$\mathcal{ALC}$, whose semantics we introduce next.

Definition 4.1 An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ consists of a non-empty *domain* $\Delta^\mathcal{I}$ and an *interpretation function* $\cdot^\mathcal{I}$ that assigns to every $a \in \mathsf{N_I}$ an element $a^\mathcal{I} \in \Delta^\mathcal{I}$, to every $A \in \mathsf{N_C}$ a fuzzy set $A^\mathcal{I}\colon \Delta^\mathcal{I} \to [0,1]$, and to every $r \in \mathsf{N_R}$ a fuzzy binary relation $r^\mathcal{I}\colon \Delta^\mathcal{I} \times \Delta^\mathcal{I} \to [0,1]$.

This function is extended to concept descriptions as follows:

- $(\neg C)^\mathcal{I}(x) = C^\mathcal{I}(x) \Rightarrow 0$,
- $(C \sqcap D)^\mathcal{I}(x) = C^\mathcal{I}(x) \otimes D^\mathcal{I}(x)$
- $(C \sqcup D)^\mathcal{I}(x) = C^\mathcal{I}(x) \oplus D^\mathcal{I}(x)$,
- $(\exists r.C)^\mathcal{I}(x) = \sup_{y \in \Delta^\mathcal{I}}(r^\mathcal{I}(x,y) \otimes C^\mathcal{I}(y))$,
- $(\forall r.C)^\mathcal{I}(x) = \inf_{y \in \Delta^\mathcal{I}}(r^\mathcal{I}(x,y) \Rightarrow C^\mathcal{I}(y))$.

We will abbreviate $A \sqcap \neg A$, where A is an arbitrary concept name, as $\bot$ and $\neg\bot$ as $\top$. It is easy to see that for every interpretation $\mathcal{I}$ and $x \in \Delta^\mathcal{I}$, we have $\bot^\mathcal{I}(x) = 0$ and $\top^\mathcal{I}(x) = 1$.

Notice that, contrary to the crisp case, existential and value restrictions are not dual, that is, in general $\neg(\exists r.\neg C)$ and $\forall r.C$ do not have the same semantics. Moreover, $\neg\neg C$ is not equivalent to C. Consider for example the Gödel t-norm, and $\mathcal{I} = (\{x\}, \cdot^\mathcal{I})$ with $A^\mathcal{I}(x) = 0.5$ and $r^\mathcal{I}(x,x) = 1$. Then (i) $(\neg A)^\mathcal{I}(x) = 0$, hence $(\neg\neg A)^\mathcal{I}(x) = 1 \neq A^\mathcal{I}(x)$, and (ii) $(\forall r.A)^\mathcal{I}(x) = r^\mathcal{I}(x,x) \Rightarrow A^\mathcal{I}(x) = 0.5$ but $(\exists r.\neg A)^\mathcal{I}(x) = 0$. Thus, we cannot assume that concept descriptions are given in negation normal form, as done for the crisp case.

The assertional and terminological knowledge is also represented by axioms, extended with a degree to which they must be satisfied.

Definition 4.2 A *fuzzy general concept inclusion (GCI)* is of the form $\langle C \sqsubseteq D \geq p\rangle$ for concept descriptions C and D and $p \in [0,1]$. A fuzzy TBox is a finite set of GCIs.

A *fuzzy assertion* is either a *concept assertion* of the form $\langle C(a) \rhd p\rangle$ or a *role assertion* of the form $\langle r(a,b) \rhd p\rangle$, where C is a concept description, $a, b \in \mathsf{N_I}$, $p \in [0,1]$, and $\rhd \in \{\geq, =\}$. A fuzzy ABox is a finite set of assertions.

An interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ *satisfies* the GCI $\langle C \sqsubseteq D \geq p\rangle$ if $C^\mathcal{I}(x) \Rightarrow D^\mathcal{I}(x) \geq p$ for all $x \in \Delta^\mathcal{I}$. It *satisfies* the assertion $\langle C(a) \rhd p\rangle$ (resp., $\langle r(a,b) \rhd p\rangle$) if $C^\mathcal{I}(a^\mathcal{I}) \rhd p$ (resp., $r^\mathcal{I}(a^\mathcal{I}, b^\mathcal{I}) \rhd p$). This interpretation is a *model* of the ABox $\mathcal{A}$ if it satisfies all assertions in $\mathcal{A}$ and of the TBox $\mathcal{T}$ if it satisfies all GCIs in $\mathcal{T}$.[7]

The ABox $\mathcal{A}$ is *consistent* w.r.t. a TBox $\mathcal{T}$ iff there is an interpretation $\mathcal{I}$ that is a model of $\mathcal{A}$ and $\mathcal{T}$.

[7] When it is clear from the context, we will often drop the prefix *fuzzy*, and speak simply of e.g. ABoxes and TBoxes.

In fuzzy DLs, reasoning is often restricted to a special kind of models, called witnessed models [18,9]. An interpretation $\mathcal{I}$ is called *witnessed* if for every concept description C, $r \in \mathsf{N_R}$, and $x \in \Delta^{\mathcal{I}}$ there exist $y, y' \in \Delta^{\mathcal{I}}$ such that

$$(\exists r.C)^{\mathcal{I}}(x) = r^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y) \text{ and } (\forall r.C)^{\mathcal{I}}(x) = r^{\mathcal{I}}(x,y') \Rightarrow C^{\mathcal{I}}(y').$$

This means that the suprema and infima in the semantics of existential and value restrictions are actually maxima and minima, respectively. Restricting to this kind of models changes the reasoning problem since there exist consistent ontologies that have no witnessed models [18]. For the rest of this paper, we consider only witnessed interpretations and models, and analyze reasoning under this restriction.

Example 4.3 Using fuzzy assertions, we can express that Chiron is human to degree 0.5 using the axiom $\langle \mathsf{Human}(\mathsf{chiron}) = 0.5 \rangle$. The fuzzy GCI $\langle \mathsf{Human} \sqsubseteq \exists \mathsf{hasParent}.\mathsf{Human} \geq 1 \rangle$ expresses that every human must have a human parent (cf. Example 2.5). This means in particular that in every model $\mathcal{I}$ of these two axioms there must be an element $x \in \Delta^{\mathcal{I}}$ such that $\mathsf{hasParent}^{\mathcal{I}}(\mathsf{chiron}^{\mathcal{I}}, x) \otimes \mathsf{Human}^{\mathcal{I}}(x) \geq 0.5$, i.e., Chiron must have a parent that is at least half human.

5 A Tableau Algorithm for $\otimes$-$\mathcal{ALC}$

In this section we describe a tableau-based approach for deciding ABox consistency. Our algorithm follows conceptually the ideas presented in [7,9], but introduces a corrected blocking condition for dealing with GCIs. To simplify the description of the method, we assume w.l.o.g. that the ABox is non-redundant in the following sense.

Definition 5.1 An ABox $\mathcal{A}$ is called *non-redundant* if it satisfies the following two conditions:

- for every concept description C and individual name a, there is at most one concept assertion of the form $\langle C(a) \triangleright p \rangle$ in $\mathcal{A}$, and
- for every role name r and individual names a, b, there is at most one role assertion of the form $\langle r(a,b) \triangleright p \rangle$ in $\mathcal{A}$.

Given $p < p' \in [0,1]$ and an ABox $\mathcal{A}$, if $\mathcal{A}$ contains $\langle C(a) = p \rangle$ and $\langle C(a) \triangleright p' \rangle$, then it is trivially inconsistent, and if $\mathcal{A}$ contains $\langle C(a) \geq p \rangle$ and $\langle C(a) \triangleright p' \rangle$, it is consistent w.r.t. a TBox $\mathcal{T}$ iff $\mathcal{A} \setminus \{\langle C(a) \geq p \rangle\}$ is consistent w.r.t. $\mathcal{T}$. A similar argument can be made for role assertions.

The following lemma shows that, under this assumption, we can consider only ABoxes where all role assertions are of the form $\langle r(a,b) = p \rangle$; i.e., no inequations appear in role assertions. This restriction will be useful for dealing with the semantics of existential and value restrictions (see Table 5.1).

Lemma 5.2 *Let $\mathcal{A}$ be an ABox such that $\mathcal{A} \cup \{\langle r(a,b) \geq p \rangle\}$ is non-redundant, and $\mathcal{T}$ a TBox. Then $\mathcal{A} \cup \{\langle r(a,b) \geq p \rangle\}$ is (witnessed-) consistent w.r.t. $\mathcal{T}$ iff $\mathcal{A} \cup \{\langle r(a,b) = p \rangle\}$ is (witnessed-) consistent w.r.t. $\mathcal{T}$.*

Proof (Sketch) Every model of $\{\langle r(a,b) = p \rangle\}$ is trivially also a model of $\{\langle r(a,b) \geq p \rangle\}$. For the converse, given a model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of $\mathcal{T}$ and $\mathcal{A} \cup \{\langle r(a,b) \geq p \rangle\}$ and $\delta \notin \Delta^{\mathcal{I}}$, we construct the interpretation $\mathcal{I}' = (\Delta^{\mathcal{I}} \cup \{\delta\}, \cdot^{\mathcal{I}'})$, where

- $b^{\mathcal{I}'} := \delta$ and $c^{\mathcal{I}'} := c^{\mathcal{I}}$ for all $c \in \mathsf{N_I} \setminus \{b\}$,

- $A^{\mathcal{I}'}(\delta) := A^{\mathcal{I}}(b^{\mathcal{I}})$ and $A^{\mathcal{I}'}(y) := A^{\mathcal{I}}(y)$ for all $A \in \mathsf{N_C}$, $y \in \Delta^{\mathcal{I}}$,
- $s^{\mathcal{I}'}(x,y) := s^{\mathcal{I}}(x,y), s^{\mathcal{I}'}(\delta,y) := s^{\mathcal{I}}(b^{\mathcal{I}},y)$ for all $s \in \mathsf{N_R}$, $x,y \in \Delta^{\mathcal{I}}$, and

$$s^{\mathcal{I}'}(x,\delta) := \begin{cases} p & \text{if } x = a^{\mathcal{I}} \text{ and } s = r, \\ s^{\mathcal{I}}(b^{\mathcal{I}}, b^{\mathcal{I}}) & \text{if } x = \delta, \\ s^{\mathcal{I}}(x, b^{\mathcal{I}}) & \text{otherwise.} \end{cases}$$

Clearly, $\mathcal{I}'$ is a model of $\{\langle r(a,b) = p\rangle\}$. It can be shown by induction on the structure of concept descriptions that, for every concept description C and every $x \in \Delta^{\mathcal{I}}$, $C^{\mathcal{I}'}(x) = C^{\mathcal{I}}(x)$ and that $C^{\mathcal{I}'}(\delta) = C^{\mathcal{I}}(b^{\mathcal{I}})$. From this, it follows that $\mathcal{I}'$ is also a model of $\mathcal{A}$ and $\mathcal{T}$. □

We now describe a tableau-based algorithm for deciding consistency of an ABox. As done for the crisp case in Section 2, we first study the case where the TBox is empty, and then extend it to deal with GCIs. Afterwards, we show that, although this extension is sound, complete, and terminating, it cannot provide a decision algorithm in general, since ABox consistency w.r.t. TBoxes is undecidable for some fuzzy DLs.

5.1 Consistency without TBoxes

The main idea underlying the tableau algorithm for fuzzy DLs is the same used for the crisp case: consistency-preserving transformation rules are used to decompose the ABox into simpler parts, until the interpretations of the atomic concepts and roles are known. This is then used to decide consistency of the input ABox. However, we cannot always specify precise degrees to the interpretations: an assertion $\langle C(a) \geq p\rangle$ simply restricts a to belong to the concept C with a degree at least p. Moreover, even if we know that the degree of, say $(C \sqcap D)(x)$ must be 0.5, we do not necessarily know the precise values for $C(x)$ and $D(x)$; the only restriction is that $C(x) \otimes D(x) = 0.5$ must hold. For example, we can choose (i) $C(x) = 1$ and $D(x) = 0.5$ or (ii) $C(x) = 0.5, D(x) = 1$ to satisfy this condition. In fact, regardless of the t-norm chosen, there are always infinitely many pairs of degrees that would satisfy this restriction. For this reason, the tableau algorithm constructs a system of constraints while decomposing the ABox.

In the following, ξ (possibly with sub- or superindices) denotes a *continuous variable* taking values from $[0,1]$, p denotes a *constant* in $[0,1]$, and ℓ denotes a *literal*, which is either a continuous variable or a constant. Recall that the tableau-based algorithm for crisp $\mathcal{ALC}$ produces a sequence of sets of ABoxes that describes the different possibilities for satisfying the restrictions from the input ABox. To generalize this idea to the fuzzy setting, we introduce the notion of *constrained ABoxes*.

Definition 5.3 A *constraint* is an expression of the form $\ell_1 \bowtie \ell$, $\ell_1 \otimes \ell_2 \bowtie \ell$, $\ell_1 \oplus \ell_2 \bowtie \ell$, or $\ell_1 \Rightarrow \ell_2 \bowtie \ell$, where ℓ_1, ℓ_2, ℓ are literals and $\bowtie \in \{\leq, =, \geq\}$.

A *constrained ABox* is a set $\mathcal{A}$ of assertions of the form $\langle C(x) \triangleright \ell\rangle$ or $\langle r(x,y) = \ell\rangle$, where C is a concept description, $r \in \mathsf{N_R}$, $\triangleright \in \{\geq, =\}$, ℓ is a literal, and x, y are individual names, such that for every individual x appearing in $\mathcal{A}$ there is a set of constraints $\mathcal{C}(x)$.

A constrained ABox can be seen as a finite representation of a possibly infinite set of ABoxes, described by the solutions of the overall constraint system $\mathfrak{C} := \bigcup_x \mathcal{C}(x)$. In the crisp case, we said that a set of ABoxes was consistent if at least one of its elements was also consistent. We generalize this idea to constrained ABoxes in the natural way: a constrained ABox $\mathcal{A}$ is *consistent* if there is a valuation of all variables in $\mathcal{A}$ that satisfies $\mathfrak{C}$ and such that the fuzzy

Table 5.1 Transformation rules for ABox consistency. ξ, ξ_1, ξ_2 are new variables.

- (A) if $\langle A(x) \triangleright \ell \rangle \in \mathcal{A}$ and $\xi_{x:A} \triangleright \ell \notin \mathcal{C}(x)$, then add $\xi_{x:A} \triangleright \ell$ to $\mathcal{C}(x)$
- (r) if $\langle r(x,y) = \ell \rangle \in \mathcal{A}$ and $\xi_{(x,y):r} = \ell \notin \mathcal{C}(y)$, then add $\xi_{(x,y):r} = \ell$ to $\mathcal{C}(y)$
- $(\sqcap)$ if $\langle C \sqcap D(x) \triangleright \ell \rangle \in \mathcal{A}$ and there are no $\langle C(x) = \chi_1 \rangle, \langle D(x) = \chi_2 \rangle \in \mathcal{A}$ with $\chi_1 \otimes \chi_2 \triangleright \ell \in \mathcal{C}(x)$, then add $\langle C(x) = \xi_1 \rangle, \langle D(x) = \xi_2 \rangle$ to $\mathcal{A}$ and $\xi_1 \otimes \xi_2 \triangleright \ell$ to $\mathcal{C}(x)$
- $(\sqcup)$ if $\langle C \sqcup D(x) \triangleright \ell \rangle \in \mathcal{A}$ and there are no $\langle C(x) = \chi_1 \rangle, \langle D(x) = \chi_2 \rangle \in \mathcal{A}$ with $\chi_1 \oplus \chi_2 \triangleright \ell \in \mathcal{C}(x)$, then add $\langle C(x) = \xi_1 \rangle, \langle D(x) = \xi_2 \rangle$ to $\mathcal{A}$ and $\xi_1 \oplus \xi_2 \triangleright \ell$ to $\mathcal{C}(x)$
- $(\neg)$ if $\langle \neg C(x) \triangleright \ell \rangle \in \mathcal{A}$ and there is no $\langle C(x) = \chi \rangle \in \mathcal{A}$ with $\chi \Rightarrow 0 \triangleright \ell \in \mathcal{C}(x)$, then add $\langle C = \xi \rangle$ to $\mathcal{A}$ and $\xi \Rightarrow 0 \triangleright \ell$ to $\mathcal{C}(x)$
- $(\exists)$ if $\langle \exists r.C(x) \triangleright \ell \rangle \in \mathcal{A}$ and there are no $\langle r(x,z) = \chi_1 \rangle, \langle C(z) = \chi_2 \rangle \in \mathcal{A}$ with $\chi_1 \otimes \chi_2 \triangleright \ell \in \mathcal{C}(z)$, then add $\langle r(x,y) = \xi_1 \rangle$ and $\langle C(y) = \xi_2 \rangle$ to $\mathcal{A}$ and $\xi_1 \otimes \xi_2 \triangleright \ell$ to $\mathcal{C}(y)$, where y is a new individual
- $(\exists')$ if $\langle \exists r.C(x) = \ell_1 \rangle, \langle r(x,y) = \ell_2 \rangle \in \mathcal{A}$ and there is no $\langle C(y) = \chi \rangle \in \mathcal{A}$ with $\ell_2 \otimes \chi \leq \ell_1 \in \mathcal{C}(y)$, then add $\langle C(y) = \xi \rangle$ to $\mathcal{A}$ and $\ell_2 \otimes \xi \leq \ell_1$ to $\mathcal{C}(y)$
- $(\forall)$ if $\langle \forall r.C(x) \triangleright \ell_1 \rangle, \langle r(x,y) = \ell_2 \rangle \in \mathcal{A}$ and there is no $\langle C(y) = \chi \rangle \in \mathcal{A}$ with $\ell_2 \Rightarrow \chi \geq \ell_1 \in \mathcal{C}(y)$, then add $\langle C(y) = \xi \rangle$ to $\mathcal{A}$ and $\ell_2 \Rightarrow \xi \geq \ell_1$ to $\mathcal{C}(y)$
- $(\forall')$ if $\langle \forall r.C(x) = \ell \rangle \in \mathcal{A}$ and there are no $\langle r(x,z) = \chi_1 \rangle, \langle C(z) = \chi_2 \rangle \in \mathcal{A}$ with $\chi_1 \Rightarrow \chi_2 = \ell \in \mathcal{C}(z)$, then add $\langle r(x,y) = \xi_1 \rangle, \langle C(y) = \xi_2 \rangle$ to $\mathcal{A}$ and $\xi_1 \Rightarrow \xi_2 = \ell$ to $\mathcal{C}(y)$, where y is a new individual

ABox obtained from $\mathcal{A}$ by replacing all variables with their valuation is consistent. To avoid an additional consistency test, we encode the latter condition into the system of constraints using auxiliary variables $\xi_{x:A}$ and $\xi_{(x,y):r}$ where x, y are individual names, A is a concept name, and r is a role name, respectively appearing in $\mathcal{A}$. Every valuation of these variables will describe an interpretation.

The algorithm starts with the input ABox $\mathcal{A}_0$ where all sets of constraints $\mathcal{C}(x)$ are empty. The transformation rules from Table 5.1 are then applied until a *complete* ABox is obtained; that is, until no rules can be applied. The main idea of these rules is that they decompose complex concepts into their subconcepts, while preserving the fuzzy semantics through the restrictions in $\mathfrak{C}$. For example, if the assertion $\langle (C \sqcap D)(x) = 0.5 \rangle$ appears in $\mathcal{A}$, the $(\sqcap)$ rule will generate two new assertions $\langle C(x) = \xi_1 \rangle$ and $\langle D(x) = \xi_2 \rangle$ with the restriction that $\xi_1 \otimes \xi_2 = 0.5$. The precise choice for the valuations of the variables ξ_1, ξ_2 will depend on the other restrictions generated by the algorithm.

As mentioned before, the rule (A) and the variable $\xi_{x:A}$ are used to ensure that every individual x obtains exactly one membership degree to the concept name A. For example, if an ABox contains the two assertions $\langle (A \sqcup B)(x) = 0.1 \rangle$ and $\langle (A \sqcap C)(x) = 0.9 \rangle$, then after application of the $(\sqcup)$ and the $(\sqcap)$ rule, the algorithm will add, among others, the two assertions $\langle A(x) = \xi \rangle$ and $\langle A(x) = \chi \rangle$, and constraints that restrict $\xi \leq 0.1$ and $\chi \geq 0.9$. Notice that both variables ξ and χ specify the membership degree of x to the concept name A, and hence this ABox is consistent only if there is a valuation that maps both variables to the same value. However, this restriction is not specified explicitly in the system of constraints. Applying the (A) rule to these assertions adds the constraints $\xi = \xi_{x:A} = \chi$, which make the system unsatisfiable. The (r) rule fulfills a similar function for role names.

When no rules can be applied, i.e., the ABox is complete, the algorithm answers "consistent" if the system $\mathfrak{C}$ is satisfiable, and "inconsistent" otherwise. Soundness and completeness of this method was shown in [9] using a slightly different syntax.

Theorem 5.4 *The following two propositions hold:*

1. *if $\mathcal{A}'$ is obtained from $\mathcal{A}$ by a rule application, then $\mathcal{A}$ is consistent if and only if $\mathcal{A}'$ is consistent;*
2. *if $\mathcal{A}$ is complete, then it is consistent iff its system of constraints $\mathfrak{C}$ is satisfiable.*

The first part of the theorem states that the transformation rules preserve consistency of the constrained ABoxes. If a sequence of rule applications reaches a complete ABox $\mathcal{A}$, then $\mathcal{A}$ is consistent iff the input ABox $\mathcal{A}_0$ is consistent. The second point of the theorem states that it suffices to test satisfiability of the system of constraints $\mathfrak{C}$ obtained to decide consistency of $\mathcal{A}$. For this method to be a decision procedure for ABox consistency, it remains to be shown that a complete ABox is obtained after finitely many rule applications, and that satisfiability of the resulting system of constraints is also decidable.

Termination holds for similar reasons as for the tableau for crisp $\mathcal{ALC}$. Every rule application adds either a new assertion or a new restriction, and removes nothing. A new concept assertion $\langle C(x) = \ell \rangle$ is added only if C is a subconcept of some D appearing in another concept assertion, to which no rule has been applied yet. Thus, the number of rule applications at a given individual is bounded by the number of occurrences of subconcepts in the input ABox $\mathcal{A}_0$. New individuals are generated by the rules $(\exists)$ and $(\forall')$. Each application of these rules generates only one new individual, and the number of such rule applications is bounded by the number of occurrences of quantified concepts in $\mathcal{A}_0$; thus, the number of successors for every individual is bounded linearly on the size of $\mathcal{A}_0$. Finally, the length of successor chains of new individuals is bounded by the maximal size of concept descriptions appearing in $\mathcal{A}_0$. This means that the ABox becomes complete after a finite number of rule applications.

The precise properties of the system of constraints depend on the t-norm $\otimes$ used for the semantics. If the Gödel t-norm is used, since the complete ABox obtained is finite, it suffices to consider only valuations that map variables to a finite set of constants. The existence of such a valuation can easily be decided e.g. by a brute-force approach testing all possibilities. Under the Łukasiewicz semantics we obtain a system of mixed integer and linear constraints [32], and under the product t-norm we get a set of (strict) quadratic constraints [7]. Satisfiability of systems of (strict) polynomial inequations is decidable in time polynomial in the number of inequations, but doubly exponential in the number of variables used [24]. This yields a decision procedure for consistency of ABoxes when the TBox is empty.

5.2 Consistency w.r.t. TBoxes

To deal with the GCIs of a TBox $\mathcal{T}$, we again extend the algorithm with the following rule:

$(\sqsubseteq)$ if $\langle C \sqsubseteq D \geq p \rangle \in \mathcal{T}$, x is an individual name in $\mathcal{A}$, and there are no $\langle C(x) = \xi_1 \rangle, \langle D(x) = \xi_2 \rangle \in \mathcal{A}$ with $\xi_1 \Rightarrow \xi_2 \geq p \in \mathcal{C}(x)$, then add $\langle C(x) = \xi_1 \rangle, \langle D(x) = \xi_2 \rangle$ to $\mathcal{A}$ and $\xi_1 \Rightarrow \xi_2 \geq p$ to $\mathcal{C}(x)$.

As in the case of crisp DLs, termination of the algorithm is not guaranteed if this rule is used. Thus, we need to develop an appropriate blocking condition that ensures termination without compromising completeness. One idea would be to directly generalize the blocking condition from Section 2 as follows: a node x is blocked by a node y if

$$\{C \mid \langle C(x) \triangleright \ell \rangle \in \mathcal{A}\} = \{C \mid \langle C(y) \triangleright \ell \rangle \in \mathcal{A}\};$$

i.e., if the sets of concepts asserted for x and y in $\mathcal{A}$ coincide. This blocking condition was proposed in [7]; however, the proof of correctness was based on the assumption that $\otimes$-$\mathcal{ALC}$ has the finite model property, which does not hold. The following example shows this, correcting a small error from [6].

Example 5.5 Under the product t-norm, consider the ABox $\mathcal{A}_0 = \{\langle \mathsf{Human}(\mathsf{chiron}) = 0.5\rangle\}$ and the TBox

$$\mathcal{T}_0 := \{\langle \top \sqsubseteq \exists \mathsf{hasParent}.\top \geq 1\rangle, \tag{5.1}$$
$$\langle \exists \mathsf{hasParent}.\mathsf{Human} \sqsubseteq \mathsf{Human} \sqcap \mathsf{Human} \geq 1\rangle, \tag{5.2}$$
$$\langle \mathsf{Human} \sqcap \mathsf{Human} \sqsubseteq \forall \mathsf{hasParent}.\mathsf{Human} \geq 1\rangle\}. \tag{5.3}$$

We construct the interpretation $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ over the domain $\Delta^{\mathcal{J}} := \{x_i \mid i \geq 0\}$ with $\mathsf{chiron}^{\mathcal{J}} := x_0$, $\mathsf{Human}^{\mathcal{J}}(x_0) := 0.5$, $\mathsf{Human}^{\mathcal{J}}(x_{i+1}) := (\mathsf{Human}^{\mathcal{J}}(x_i))^2$ for $i \geq 0$, and

$$\mathsf{hasParent}^{\mathcal{J}}(x_i, x_j) := \begin{cases} 1 & \text{if } i+1 = j \\ 0 & \text{otherwise.} \end{cases}$$

This interpretation is a model of $\mathcal{A}_0$. For every $x_i \in \Delta^{\mathcal{I}}$, $\mathsf{hasParent}^{\mathcal{I}}(x_i, x_{i+1}) = 1$ holds, and hence $\mathcal{J}$ satisfies the first axiom in $\mathcal{T}_0$. Additionally,

$$\begin{aligned} (\exists \mathsf{hasParent}.\mathsf{Human})^{\mathcal{J}}(x_i) &= (\forall \mathsf{hasParent}.\mathsf{Human})^{\mathcal{J}}(x_i) \\ &= \mathsf{Human}^{\mathcal{J}}(x_{i+1}) \\ &= (\mathsf{Human}^{\mathcal{J}}(x_i))^2 \\ &= (\mathsf{Human} \sqcap \mathsf{Human})^{\mathcal{J}}(x_i), \end{aligned}$$

which implies that $\mathcal{J}$ is also a model of the last two axioms in $\mathcal{T}_0$.

It is easy to prove by induction that for every witnessed model $\mathcal{I}$ of $\mathcal{A}_0$ and $\mathcal{T}_0$, there must exist elements $y_0, y_1, \ldots \in \Delta^{\mathcal{I}}$ such that $\mathsf{Human}^{\mathcal{I}}(y_i) = \mathsf{Human}^{\mathcal{J}}(x_i)$. Since we know that $\mathsf{Human}^{\mathcal{J}}(x_i) \neq \mathsf{Human}^{\mathcal{J}}(x_j)$ for all $i \neq j$, it follows that $\mathcal{I}$ must also have an infinite domain. Thus there is no finite model of $\mathcal{A}_0$ and $\mathcal{T}_0$.

In fact, if this blocking condition is used, the algorithm might not detect some inconsistencies, as shown in [3].

Example 5.6 Consider $\mathcal{A}_0$ from Example 5.5 and $\mathcal{T}_1 := \mathcal{T}_0 \cup \{\langle \top \sqsubseteq \mathsf{Human} \geq 0.05\rangle\}$. It follows from the arguments of Example 5.5 that $\mathcal{A}_0$ is inconsistent w.r.t. $\mathcal{T}_1$.

The tableau algorithm produces only two new individuals x_1, x_2 and restrictions that enforce that the degrees of Human for x_1 and x_2 must be 0.25 and 0.0625, respectively. At this point, the blocking condition is triggered, and no inconsistency has been found, hence the algorithm answers that $\mathcal{A}_0$ is consistent w.r.t. $\mathcal{T}_1$.[8]

Despite the lack of the finite model property, it is still possible to define a blocking condition that preserves correctness. The idea is that, rather than blocking a node if it considers the same concept descriptions as a predecessor, we require the existence of an "isomorphism", in the following sense, between their systems of constraints.

Definition 5.7 For a set of constraints $\mathcal{C}$, let $\mathsf{var}(\mathcal{C})$ denote the variables appearing in $\mathcal{C}$. An *isomorphism* between two sets of constraint $\mathcal{C}, \mathcal{C}'$ is a bijective function $f\colon \mathsf{var}(\mathcal{C}) \to \mathsf{var}(\mathcal{C}')$ such that for every constraint $\mathsf{c}(\ell_1, \ldots, \ell_n)$ it holds that

$$\mathsf{c}(\ell_1, \ldots, \ell_n) \in \mathcal{C} \quad \text{iff} \quad \mathsf{c}(\hat{f}(\ell_1), \ldots, \hat{f}(\ell_n)) \in \mathcal{C}',$$

where ℓ_i, $1 \leq i \leq n$, are literals from $\mathcal{C}$ and

$$\hat{f}(\ell) = \begin{cases} \ell & \text{if } \ell \text{ is a constant,} \\ f(\ell) & \text{otherwise.} \end{cases}$$

[8] For full details, see [3].

The blocking condition is defined in terms of isomorphisms between the systems of constraints $\mathcal{C}(x)$ and $\mathcal{C}(y)$ of the individuals x, y. Although this will not define a finite (cyclic) model, it yields a finite description of the infinite system of constraints that would be produced by the algorithm without blocking.

Definition 5.8 Two nodes x, y are *equivalent*, denoted as $x \approx y$, if there exists an isomorphism f between $\mathcal{C}(x)$ and $\mathcal{C}(y)$ such that for every concept C and literal ℓ in $\mathcal{C}(x)$, we have $\langle C(x) \triangleright \ell \rangle \in \mathcal{A}$ iff $\langle C(y) \triangleright \hat{f}(\ell) \rangle \in \mathcal{A}$.

A node x is *directly blocked* iff it is not a root node and it has an ancestor y that is not a root node such that $x \approx y$; in this case we say that y is the *blocking* node of x. A non-root node x is *blocked* if it is directly blocked or its (unique) predecessor is blocked. In the latter case, we say that x is *indirectly* blocked.

When a node is (directly or indirectly) blocked, then none of the rules $(\exists)$ or $(\forall')$ may be applied, which disallows the creation of new individuals. Notice that none of the other rules produce any new individuals or edges; they only decompose the information contained in the respective node to simpler concepts. As the following lemma shows, this notion suffices for obtaining a terminating procedure.

Lemma 5.9 *The algorithm terminates if the rules $(\exists)$ and $(\forall')$ are not applied to any assertion $\langle \exists r.C(x) \triangleright \ell \rangle$ (resp. $\langle \forall r.C(x) = \ell \rangle$) where x is blocked.*

Proof As for the case with empty TBoxes, the number of rule applications at any given node and the number of successors of every node are bounded polynomially in the size of the input ABox and TBox.

It only remains to show that a blocked node is found eventually on each branch and hence there is no infinite chain of successor individuals. Once again, every node may contain at most one assertion for each occurrence of a concept description in the input ABox and TBox. A restriction is added to $\mathcal{C}$ only if an assertion has been added. This together implies that every chain of successor nodes eventually contains a node that is blocked. Thus the algorithm terminates. □

When the algorithm terminates, the ABox it has produced can be seen as a finite *forest*, i.e. a set of trees arbitrarily interconnected at their roots, where every leaf is blocked or contains no quantified assertions. We can prune this forest in such a way that every leaf is *directly* blocked. We call this the *pruned forest*. This forest has an associated finite system of constraints. We can then "unravel" this system into an infinite one that characterizes ontology consistency.

Let $\mathcal{A}$ be a pruned forest with n leaves and block a function that maps every directly blocked node to its blocking node, and all other leaves to themselves. Note that for every leaf x, both it and $\mathsf{block}(x)$ belong to the same tree-like substructure of $\mathcal{A}$ since $\mathsf{block}(x)$ cannot be a root node. For every leaf x_i, $1 \leq i \leq n$, let $\mathcal{A}(i)$ be the sub-ABox of $\mathcal{A}$ that contains only the individuals at the subtree with root $\mathsf{block}(x_i)$ and $\mathfrak{C}(i)$ the set of all restrictions appearing in $\mathcal{A}(i)$. We define the binary relation $\mapsto \subseteq \{1,\dots,n\} \times \{1,\dots,n\}$ where $i \mapsto j$ iff the leaf x_j is a successor of $\mathsf{block}(x_i)$. Finally, we define the language L_t as the smallest set of words in $\{1,\dots,n\}^*$ that contains $\{\varepsilon, 1,\dots,n\}$ and such that if $\eta i \in L_t$ and $i \mapsto j$, then also $\eta i j \in L_t$.

Definition 5.10 For every word $\eta i \in L_t$, let $\mathcal{A}_{\eta i}$ and $\mathfrak{C}_{\eta i}$ be disjoint copies of $\mathcal{A}(i)$ and $\mathfrak{C}(i)$, respectively, where each individual x in $\mathcal{A}(i)$ is renamed to $x^{\eta i}$ and each variable ξ in $\mathfrak{C}(i)$ is renamed to $\xi^{\eta i}$. We define $\mathcal{A}_{\mathcal{O}}$ as the (infinite) constrained ABox that contains $\mathcal{A}_\varepsilon := \mathcal{A}$ with constraints $\mathfrak{C}_\varepsilon := \mathfrak{C}$, and for every $\eta i \in L_t$ contains the ABox $\mathcal{A}_{\eta i}$ with $(\mathsf{block}(x_i))^{\eta i}$ renamed

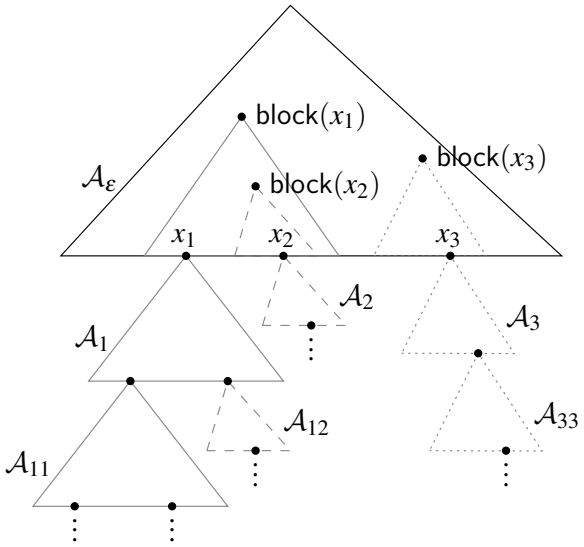

Fig. 5.1 Construction of $\mathcal{A}_\mathcal{O}$.

to x_i^η and the constraint system $\mathfrak{C}_{\eta i}$ with the additional restriction $\xi^\eta = f(\xi)^{\eta i}$ for each variable ξ of $\mathcal{C}(x_i)$, where f is the isomorphism between $\mathcal{C}(x_i)$ and $\mathcal{C}(\mathsf{block}(x_i))$ provided by the blocking condition. The infinite system of constraints of $\mathcal{A}_\mathcal{O}$ is denoted by $\mathfrak{C}_\mathcal{O}$.

Intuitively, the infinite ABox $\mathcal{A}_\mathcal{O}$ is an abstract description of a model of the input ABox $\mathcal{A}_0$ and TBox $\mathcal{T}$ which is obtained by applying the tableau rules without the blocking condition. Every (directly) blocked node x_i stores information that is isomorphic to that appearing in $\mathsf{block}(x_i)$. If we were to apply the $(\exists)$ and $(\forall')$ rules to it, we would obtain an isomorphic copy of $\mathcal{A}(i)$ with its corresponding system of constraints $\mathfrak{C}(i)$, as depicted in Figure 5.1. In other words, the pruned forest $\mathcal{A}$ can be seen as a finite representation of the infinite ABox $\mathcal{A}_\mathcal{O}$, which is obtained by appending finite sub-trees at every leaf node.

To decide consistency, one still needs to check the system $\mathfrak{C}_\mathcal{O}$ for satisfiability, i.e., whether we can instantiate the variables in $\mathcal{A}_\mathcal{O}$ in such a way that we obtain an actual model of $\mathcal{A}_0$ and $\mathcal{T}$. The correctness of this approach can be shown similarly to the correctness of the tableau algorithm without GCIs. A model $\mathcal{I}$ of $\mathcal{A}_0$ and $\mathcal{T}$ can be used to find a solution of the system of inequations $\mathfrak{C}_\mathcal{O}$, while a solution to this system informs the construction of a model $\mathcal{I}$. This is formalized in the following theorem.

Theorem 5.11 *Let $\mathfrak{C}_\mathcal{O}$ be the system of constraints obtained from the pruned output forest of the algorithm applied to an ABox $\mathcal{A}_0$ and a TBox $\mathcal{T}$. Then $\mathfrak{C}_\mathcal{O}$ is satisfiable iff $\mathcal{A}_0$ is consistent w.r.t. $\mathcal{T}$.*

Proof Let $\mathcal{A}_\mathcal{O}$ be the ABox from which the system $\mathfrak{C}_\mathcal{O}$ was obtained. By construction, every variable appearing in $\mathfrak{C}_\mathcal{O}$ is either (i) of the form $\xi_{x:A}$ or $\xi_{(x,y):r}$, or (ii) corresponds to exactly one assertion of the form $\langle C(x) = \xi \rangle$ or $\langle r(x,y) = \xi \rangle$ in $\mathcal{A}_\mathcal{O}$.

Suppose that $\mathfrak{C}_\mathcal{O}$ is satisfiable, and let $\mathcal{V}$ be a valuation that satisfies all the constraints. We define the interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ where $\Delta^\mathcal{I} := \{x \mid \xi_{x:A}$ is a variable in $\mathfrak{C}_\mathcal{O}, A \in \mathsf{N_C}\}$,

$x^{\mathcal{I}} := x$ for every individual name x, and

$$A^{\mathcal{I}}(x) := \begin{cases} \mathcal{V}(\xi_{x:A}) & \text{if } \xi_{x:A} \text{ appears in } \mathfrak{C}_{\mathcal{O}} \\ 0 & \text{otherwise;} \end{cases}$$

$$r^{\mathcal{I}}(x,y) := \begin{cases} \mathcal{V}(\xi_{(x,y):r}) & \text{if } \xi_{(x,y):r} \text{ appears in } \mathfrak{C}_{\mathcal{O}} \\ 0 & \text{otherwise.} \end{cases}$$

It is easy to show by induction on the structure of the concept descriptions that for every individual x appearing in $\mathcal{A}_{\mathcal{O}}$, if $\langle C(x) = \xi \rangle \in \mathcal{A}_{\mathcal{O}}$, then $C^{\mathcal{I}}(x) = \mathcal{V}(\xi)$. Since $\mathcal{V}$ satisfies $\mathfrak{C}_{\mathcal{O}}$, for every $\langle C(a) \triangleright p \rangle \in \mathcal{A}$ it follows that $C^{\mathcal{I}}(a) \triangleright p$, and hence $\mathcal{I}$ is a model of $\mathcal{A}_0 \subseteq \mathcal{A}$. Let now $\langle C \sqsubseteq D \geq p \rangle \in \mathcal{T}$. For every individual $x \in \Delta^{\mathcal{I}}$, there are two assertions $\langle C(x) = \xi_1 \rangle$ and $\langle C(x) = \xi_2 \rangle$ in $\mathcal{A}_{\mathcal{O}}$ with the restriction $\xi_1 \Rightarrow \xi_2 \geq p$ in $\mathfrak{C}_{\mathcal{O}}$. From this, it follows that $C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x) = \mathcal{V}(\xi_1) \Rightarrow \mathcal{V}(\xi_2) \geq p$, and hence $\mathcal{I}$ is a model of $\mathcal{T}$.

Conversely, let $\mathcal{I}$ be a model of $\mathcal{A}_0$ and $\mathcal{T}$. We use this interpretation to build a valuation $\mathcal{V}$ that satisfies $\mathfrak{C}_{\mathcal{O}}$ and for which $\mathcal{I}$ satisfies all assertions in $\mathcal{A}_{\mathcal{O}}$, i.e., we have $C^{\mathcal{I}}(x^{\mathcal{I}}) \triangleright \mathcal{V}(\ell)$ whenever $\langle C(x) \triangleright \ell \rangle \in \mathcal{A}_{\mathcal{O}}$, and similarly for role assertions. We proceed inductively on the rule applications. Whenever a rule introduces a new individual, we have to appropriately extend $\mathcal{I}$ while ensuring that it still satisfies $\mathcal{T}$.

At the start of the tableau algorithm, for every concept assertion $\langle C(a) \triangleright p \rangle \in \mathcal{A}_0$ we have $C^{\mathcal{I}}(a^{\mathcal{I}}) \triangleright p = \mathcal{V}(p)$ since $\mathcal{I}$ is a model of $\mathcal{A}_0$; likewise for role assertions. Let now $\langle C(x) \triangleright \ell \rangle \in \mathcal{A}_{\mathcal{O}}$ be such that $C^{\mathcal{I}}(x^{\mathcal{I}}) \triangleright \mathcal{V}(\ell)$.

If C is of the form $C_1 \sqcap C_2$, we have $\langle C_1(x) = \xi_1 \rangle, \langle C_2(x) = \xi_2 \rangle$ in $\mathcal{A}_{\mathcal{O}}$ and $\xi_1 \otimes \xi_2 \triangleright \ell$ in $\mathfrak{C}_{\mathcal{O}}$ since $\mathcal{A}_{\mathcal{O}}$ was constructed from the complete ABox $\mathcal{A}_{\varepsilon}$. Setting $\mathcal{V}(\xi_1) := C_1^{\mathcal{I}}(x^{\mathcal{I}})$ and $\mathcal{V}(\xi_2) := C_2^{\mathcal{I}}(x^{\mathcal{I}})$ satisfies the two assertions and the constraint.

If C is of the form $\exists r.D$, by assumption $C^{\mathcal{I}}(x^{\mathcal{I}}) \triangleright \mathcal{V}(\ell)$ and hence, there is a $y_0 \in \Delta^{\mathcal{I}}$ such that $r^{\mathcal{I}}(x^{\mathcal{I}}, y_0) \otimes D^{\mathcal{I}}(y_0) \triangleright \mathcal{V}(\ell)$. If x is not a copy of a blocked node in $\mathcal{A}_{\varepsilon}$, then completeness of $\mathcal{A}_{\varepsilon}$ ensures the existence of an individual name y such that $\langle r(x,y) = \xi_1 \rangle, \langle D(y) = \xi_2 \rangle \in \mathcal{A}_{\varepsilon}$ and $\xi_1 \otimes \xi_2 \triangleright \ell \in \mathfrak{C}_{\mathcal{O}}$. Setting $y^{\mathcal{I}} := y_0$, $\mathcal{V}(\xi_1) := r^{\mathcal{I}}(x^{\mathcal{I}}, y_0)$, and $\mathcal{V}(\xi_2) := D^{\mathcal{I}}(y_0)$ satisfies the claim. If x is a copy of the blocked individual x_i in the ABox $\mathcal{A}_{\eta}$, then there is an individual name y and variables ξ_1, ξ_2 with $\langle r(x, y^{\eta i}) = \xi_1^{\eta i} \rangle, \langle D(y^{\eta i}) = \xi_2^{\eta i} \rangle \in \mathcal{A}_{\mathcal{O}}$ and $\mathfrak{C}_{\mathcal{O}}$ contains the restriction $\xi_1^{\eta i} \otimes \xi_2^{\eta i} \triangleright \ell$. Again, setting $(y^{\eta i})^{\mathcal{I}} := y_0$, $\mathcal{V}(\xi_1^{\eta i}) := r^{\mathcal{I}}(x^{\mathcal{I}}, y_0)$, and $\mathcal{V}(\xi_2^{\eta i}) = D^{\mathcal{I}}(y_0)$ satisfies the claim.

All other rules can be treated analogously. This procedure inductively defines a valuation satisfying all constraints in $\mathfrak{C}_{\mathcal{O}}$. □

We thus have a terminating algorithm that produces a finite representation of an infinite constraint system. The algorithm is sound and complete for deciding consistency in the sense that the generated system is satisfiable iff the input ABox is consistent w.r.t. the TBox. However, it is not known how to decide satisfiability of an infinite system of constraints, even if it is finitely represented as $\mathfrak{C}_{\varepsilon}$. As we show in the following section, this problem is in fact undecidable for some t-norms.

The above construction is nevertheless useful since it illustrates that the cause of undecidability does not lie in the blocking condition, but rather in the inability to detect inconsistencies in the infinite constraint systems described by $\mathfrak{C}_{\varepsilon}$. If one could identify a sublogic of $\otimes$-$\mathcal{ALC}$ for which satisfiability of the resulting constraints is decidable, our tableau algorithm would immediately yield a decision procedure for consistency in this logic.

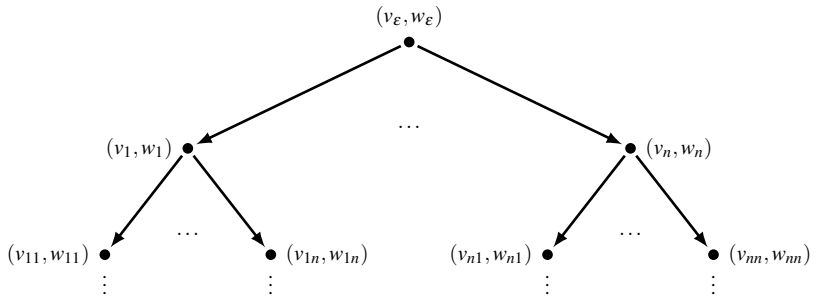

Fig. 6.1 An instance of the PCP.

6 Undecidable Fuzzy DLs

The consistency problem w.r.t. general TBoxes has been recently shown to be undecidable for several fuzzy DLs [3,4,13,15]. While these undecidability proofs differ in their details, they are all based on the same general idea. In this section, we provide a prototypical proof of undecidability by showing that witnessed consistency of Π-$\mathcal{ALC}$ ABoxes w.r.t. TBoxes is undecidable. This proof is intended to highlight the core steps followed by most other undecidability proofs for fuzzy DLs.

For this whole section, the t-norm is always assumed to be the product t-norm. To show undecidability, we use a reduction from a slight variant of the Post correspondence problem, which is well-known to be undecidable [27].

Definition 6.1 Let $(v_1, w_1), \ldots, (v_m, w_m)$ be a finite list of pairs of words over an alphabet $\Sigma = \{1, \ldots, s\}, s > 1$. The *Post correspondence problem* (PCP) asks whether there is a sequence $i_1, i_2, \ldots, i_k$, $1 \leq i_j \leq m$, such that $v_1 v_{i_1} v_{i_2} \cdots v_{i_k} = w_1 w_{i_1} w_{i_2} \cdots w_{i_k}$. If such a sequence exists, the word $i_1 i_2 \cdots i_k$ is called a *solution* of the problem.

For a word $\mu = i_1 i_2 \cdots i_k \in \{1, \ldots, m\}^*$, we will denote as v_μ and w_μ the words $v_1 v_{i_1} v_{i_2} \cdots v_{i_k}$ and $w_1 w_{i_1} w_{i_2} \cdots w_{i_k}$, respectively. Intuitively, we can see every instance of the PCP as an m-ary infinite tree such that (i) the root node is labeled with $(v_\varepsilon, w_\varepsilon) = (v_1, w_1)$, and (ii) if a node is labeled with the pair (v, w), then its i-th successor is labeled with the pair (vv_i, ww_i) (see Figure 6.1). The PCP then consists in deciding whether there is a node in this tree whose label (v, w) is such that $v = w$.

Recall that the alphabet Σ consists of the first s positive integers. We can thus view every word in Σ^* as a natural number represented in base $s+1$ in which 0 never occurs. Slightly abusing this intuition, we will express the empty word as the number 0. In the following reductions, we encode the word u in Σ^* using the number $2^{-u} \in [0, 1]$.

The main idea of the reduction is to construct an ABox and a TBox whose models encode an instance of the PCP, viewed as a tree. In other words, every model contains, for each possible solution $\mu \in \{1, \ldots, m\}^*$, a domain element at which the interpretation of two designated concept names A and B will correspond to the words v_μ, w_μ, respectively. More formally, for a given instance $\mathcal{P} = ((v_1, w_1), \ldots, (v_m, w_m))$ of the PCP, we define an ABox $\mathcal{A}_\mathcal{P}$ and a TBox $\mathcal{T}_\mathcal{P}$ such that for every witnessed model $\mathcal{I}$ of $\mathcal{A}_\mathcal{P}$ and $\mathcal{T}_\mathcal{P}$ and every $\mu \in \{1, \ldots, m\}^*$ there is an element $\delta_\mu \in \Delta^\mathcal{I}$ with $A^\mathcal{I}(\delta_\mu) = 2^{-v_\mu}$ and $B^\mathcal{I}(\delta_\mu) = 2^{-w_\mu}$. Additionally, we will show that there is a witnessed model $\mathcal{I}_\mathcal{P}$ of $\mathcal{A}_\mathcal{P}$ and $\mathcal{T}_\mathcal{P}$ whose domain has only these elements (see Figure 6.2). Then, $\mathcal{P}$ has a solution iff for every witnessed

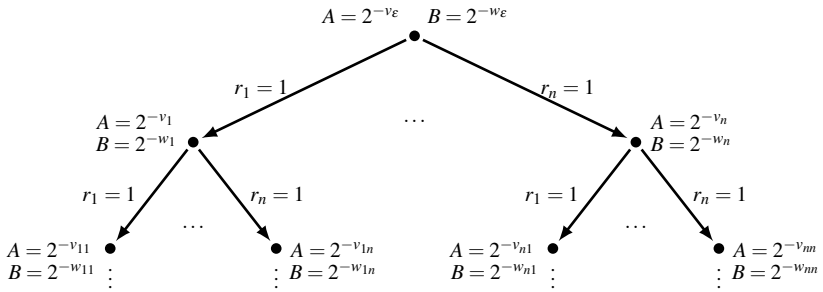

Fig. 6.2 The interpretation $\mathcal{I}_\mathcal{P}$.

model $\mathcal{I}$ of $\mathcal{A}_\mathcal{P}$ and $\mathcal{T}_\mathcal{P}$ there exists a $\delta \in \Delta^\mathcal{I}$ such that $A^\mathcal{I}(\delta) = B^\mathcal{I}(\delta)$.

We first introduce some abbreviations that will be helpful in simplifying the construction. We use the expression C^n to denote the n-ary conjunction of a concept description C with itself; formally, $C^0 := \top$ and $C^{n+1} := C \sqcap C^n$ for every $n \geq 0$. Given two concept descriptions C, D and a role name r, we use the expression $\langle C \overset{r}{\rightsquigarrow} D \rangle$ to denote the two axioms $\langle C \sqsubseteq \forall r.D \geq 1 \rangle$ and $\langle \exists r.D \sqsubseteq C \geq 1 \rangle$. These axioms are used to transfer specific values along role connections.

Lemma 6.2 *For every interpretation $\mathcal{I}$ and all $x, y \in \Delta^\mathcal{I}$, we have*

- $(C^n)^\mathcal{I}(x) = (C^\mathcal{I}(x))^n$*; and*
- *if $\mathcal{I}$ satisfies $\langle C \overset{r}{\rightsquigarrow} D \rangle$ and $r^\mathcal{I}(x,y) = 1$, then $C^\mathcal{I}(x) = D^\mathcal{I}(y)$.*

Proof The first equality is obvious from the definition of C^n and the fact that we are using the product t-norm. For the second equality, since $\mathcal{I}$ satisfies $\langle C \sqsubseteq \forall r.D \geq 1 \rangle$, we have $C^\mathcal{I}(x) \leq (\forall r.D)^\mathcal{I}(x)$, and thus

$$C^\mathcal{I}(x) \leq \inf_{\delta \in \Delta^\mathcal{I}} (r^\mathcal{I}(x,\delta) \Rightarrow D^\mathcal{I}(\delta)) \leq r^\mathcal{I}(x,y) \Rightarrow D^\mathcal{I}(y) = D^\mathcal{I}(y).$$

Dually, since $\mathcal{I}$ satisfies $\langle \exists r.D \sqsubseteq C \geq 1 \rangle$, we know that $(\exists r.D)^\mathcal{I}(x) \leq C^\mathcal{I}(x)$, and hence

$$C^\mathcal{I}(x) \geq \sup_{\delta \in \Delta^\mathcal{I}} (r^\mathcal{I}(x,\delta) \otimes D^\mathcal{I}(\delta)) \geq r^\mathcal{I}(x,y) \otimes D^\mathcal{I}(y) = D^\mathcal{I}(y).$$

These two facts together imply $C^\mathcal{I}(x) = D^\mathcal{I}(y)$, as claimed. □

We are now ready to construct the search space for a solution of $\mathcal{P}$ as depicted in Figure 6.2. We do this step-wise through the following subsections. We first show how to enforce the tree-like structure in every model. Afterwards, we add new axioms that allow us to detect whether a solution exists or not.

6.1 Constructing the Successor Nodes

Assume first that we have already constructed a node $\delta \in \Delta^\mathcal{I}$ in some interpretation that encodes the words $v, w \in \Sigma^*$, i.e., we have $A^\mathcal{I}(\delta) = 2^{-v}$ and $B^\mathcal{I}(\delta) = 2^{-w}$. Our goal is to ensure, for each $i, 1 \leq i \leq m$, the existence of an r_i-successor of δ that encodes the concatenation of the words v, w with the i-th pair from $\mathcal{P}$, i.e., vv_i and ww_i. We assume for

now that we also have concept names V_i, W_i that encode v_i and w_i, respectively, i.e., we know that $V_i^{\mathcal{I}}(\delta) = 2^{-v_i}$ and $W_i^{\mathcal{I}}(\delta) = 2^{-w_i}$. We use the TBox

$$\mathcal{T}_{\mathcal{P}}^i := \{\langle \top \sqsubseteq \exists r_i.\top \geq 1\rangle,\ \langle (V_i \sqcap A^{(s+1)^{|v_i|}}) \overset{r_i}{\rightsquigarrow} A\rangle,\ \langle (W_i \sqcap B^{(s+1)^{|w_i|}}) \overset{r_i}{\rightsquigarrow} B\rangle\}.$$

Recall that we are viewing words in Σ^* as natural numbers in base $s+1$. Thus, the concatenation of two words u, u' corresponds to the operation $u \cdot (s+1)^{|u'|} + u'$ on natural numbers. We thus have

$$(V_i \sqcap A^{(s+1)^{|v_i|}})^{\mathcal{I}}(\delta) = V_i^{\mathcal{I}}(\delta) \cdot (A^{\mathcal{I}}(\delta))^{(s+1)^{|v_i|}} = 2^{-(v(s+1)^{|v_i|} + v_i)} = 2^{-vv_i}.$$

If $\mathcal{I}$ is a witnessed model of $\mathcal{T}_{\mathcal{P}}^i$, then from the first axiom it follows that there must exist an element $\gamma \in \Delta^{\mathcal{I}}$ with $r^{\mathcal{I}}(\delta, \gamma) = 1$. The last two axioms then ensure that $A^{\mathcal{I}}(\gamma) = 2^{-vv_i}$ and $B^{\mathcal{I}}(\gamma) = 2^{-ww_i}$; thus, the concept names A and B encode, at node γ, the words vv_i and ww_i, as desired.

If we want to use this construction to recursively produce all the pairs of concatenated words defined by $\mathcal{P}$, we need to ensure also that $V_j^{\mathcal{I}}(\gamma) = 2^{-v_j}$ and $W_j^{\mathcal{I}}(\gamma) = 2^{-w_j}$ hold for every $j, 1 \leq j \leq m$. This can be done through the axioms

$$\mathcal{T}_{\mathcal{P}}^0 := \{\langle V_j \overset{r_i}{\rightsquigarrow} V_j\rangle, \langle W_j \overset{r_i}{\rightsquigarrow} W_j\rangle \mid 1 \leq i, j \leq m\}.$$

6.2 Constructing the Root Node

We now need to ensure that there is a node δ_ε with $A^{\mathcal{I}}(\delta_\varepsilon) = 2^{-v_1}$ and $B^{\mathcal{I}}(\delta_\varepsilon) = 2^{-w_1}$ (that is, where A and B encode v_ε and w_ε, respectively) and where $V_j^{\mathcal{I}}(\delta_\varepsilon) = 2^{-v_j}$, $W_j^{\mathcal{I}}(\delta_\varepsilon) = 2^{-w_j}$ hold for every $j, 1 \leq j \leq m$. This condition is easily enforced through the ABox[9]

$$\begin{aligned}\mathcal{A}_{\mathcal{P}}^0 := {} & \{\langle A(a) = 2^{-v_1}\rangle, \langle B(a) = 2^{-w_1}\rangle\} \cup \\ & \{\langle V_i(a) = 2^{-v_i}\rangle, \langle W_i(a) = 2^{-w_i}\rangle \mid 1 \leq i \leq m\}.\end{aligned}$$

Clearly, every model $\mathcal{I}$ of the ABox $\mathcal{A}_{\mathcal{P}}^0$ contains an element $\delta_\varepsilon := a^{\mathcal{I}}$ that represents the root of the tree.

6.3 Auxiliary Concept Names

In the previous two subsections, we have introduced a set of axioms that can only be satisfied by models that encode the instance $\mathcal{P}$ of the PCP, as depicted in Figure 6.2. We now turn our attention to deciding whether this instance has a solution; i.e., whether there is a node in this tree that satisfies the concepts A and B to the same degree. To achieve this, we include two auxiliary concept names H and X. The former will be interpreted as 0.5 in every domain element, while the latter will be a *crisp* concept; that is, $X^{\mathcal{I}}(x) \in \{0, 1\}$ for every $x \in \Delta^{\mathcal{I}}$. These two properties are enforced by the following axioms:

$$\begin{aligned}\mathcal{A}_0 &:= \{\langle H(a) = 0.5\rangle\}, \\ \mathcal{T}_0 &:= \{\langle H \overset{r_i}{\rightsquigarrow} H\rangle \mid 1 \leq i \leq m\} \cup \{\langle X \sqsubseteq X \sqcap X \geq 1\rangle\}.\end{aligned}$$

[9] Notice that equality is necessary for this construction; it is not possible to express $\langle X(a) = q\rangle$ with $q < 1$ using only axioms of the form $\langle Y(a) \geq q'\rangle$.

The last axiom expresses that $X^{\mathcal{I}}(x) \leq X^{\mathcal{I}}(x) \cdot X^{\mathcal{I}}(x)$, and hence $X^{\mathcal{I}}(x) \in \{0,1\}$ for every x. These concept names will later be used to detect whether $\mathcal{P}$ has a solution (see Theorem 6.5), but first we prove that the interpretation $\mathcal{I}_{\mathcal{P}}$ from Figure 6.2 is indeed a model of all these axioms, and moreover, that every model must *include* $\mathcal{I}_{\mathcal{P}}$ as formalized next.

6.4 The Canonical Model

Let $\mathcal{A}_{\mathcal{P}} := \mathcal{A}_0 \cup \mathcal{A}^0_{\mathcal{P}}$ and $\mathcal{T}_{\mathcal{P}} := \mathcal{T}_0 \cup \bigcup_{i=0}^{m} \mathcal{T}^i_{\mathcal{P}}$. We define the interpretation $\mathcal{I}_{\mathcal{P}} := (\Delta^{\mathcal{I}_{\mathcal{P}}}, \cdot^{\mathcal{I}_{\mathcal{P}}})$ as follows:

- $\Delta^{\mathcal{I}_{\mathcal{P}}} = \{1,\dots,m\}^*$,
- $a^{\mathcal{I}_{\mathcal{P}}} = \varepsilon$,

for every $\mu \in \Delta^{\mathcal{I}_{\mathcal{P}}}$,

- $A^{\mathcal{I}_{\mathcal{P}}}(\mu) = 2^{-v_\mu}$, $B^{\mathcal{I}_{\mathcal{P}}}(\mu) = 2^{-w_\mu}$,
- $H^{\mathcal{I}_{\mathcal{P}}}(\mu) = 0.5$,
- $X^{\mathcal{I}_{\mathcal{P}}}(\mu) = \begin{cases} 1 \text{ if } A^{\mathcal{I}_{\mathcal{P}}}(\mu) \leq B^{\mathcal{I}_{\mathcal{P}}}(\mu), \\ 0 \text{ otherwise,} \end{cases}$

and for all $j, 1 \leq j \leq m$

- $V_j^{\mathcal{I}_{\mathcal{P}}}(\mu) = 2^{-v_j}$, $W_j^{\mathcal{I}_{\mathcal{P}}}(\mu) = 2^{-w_j}$, and
- $r_j^{\mathcal{I}_{\mathcal{P}}}(\mu, \mu j) = 1$ and $r_j^{\mathcal{I}_{\mathcal{P}}}(\mu, \mu') = 0$ if $\mu' \neq \mu j$.

This interpretation is precisely the one depicted in Figure 6.2; the figure does not depict interpretation of the concept H as it is always the constant 0.5. Notice that the concept X is interpreted as the Boolean variable deciding whether A is smaller or equal to B or not, at every node. It is easy to see that $\mathcal{I}_{\mathcal{P}}$ is in fact a witnessed model of $\mathcal{A}_{\mathcal{P}}$ and $\mathcal{T}_{\mathcal{P}}$, since every node has exactly one r_i-successor with degree greater than 0, for every $i, 1 \leq i \leq m$. More interesting, however, is that for every witnessed model $\mathcal{I}$ of $\mathcal{A}_{\mathcal{P}}$ and $\mathcal{T}_{\mathcal{P}}$, there is a homomorphism from $\mathcal{I}_{\mathcal{P}}$ to $\mathcal{I}$ as described in the following lemma.

Lemma 6.3 *Let $\mathcal{I}$ be a witnessed model of $\mathcal{A}_{\mathcal{P}}$ and $\mathcal{T}_{\mathcal{P}}$. There is a function $f\colon \Delta^{\mathcal{I}_{\mathcal{P}}} \to \Delta^{\mathcal{I}}$ such that, for every $\mu \in \Delta^{\mathcal{I}_{\mathcal{P}}}$ and every concept name C appearing in $\mathcal{A}_{\mathcal{P}}$ or $\mathcal{T}_{\mathcal{P}} \setminus \mathcal{T}_0$, $C^{\mathcal{I}_{\mathcal{P}}}(\mu) = C^{\mathcal{I}}(f(\mu))$ holds.*

Proof The function f is built inductively on the length of μ. For $\mu = \varepsilon$, notice that $\mathcal{A}_{\mathcal{P}}$ fixes the interpretation of all relevant concept names at $a^{\mathcal{I}}$ and hence defining $f(\varepsilon) := a^{\mathcal{I}}$ satisfies the condition of the lemma.

Let now μ be such that $f(\mu)$ has already been defined. By induction, we assume that $A^{\mathcal{I}}(f(\mu)) = 2^{-v_\mu}$, $B^{\mathcal{I}}(f(\mu)) = 2^{-w_\mu}$, $H^{\mathcal{I}}(f(\mu)) = 0.5$, and for every j, $1 \leq j \leq m$, $V_j^{\mathcal{I}}(f(\mu)) = 2^{-v_j}, W_j^{\mathcal{I}}(f(\mu)) = 2^{-w_j}$. Since $\mathcal{I}$ is a witnessed model of $\langle \top \sqsubseteq \exists r_i.\top \geq 1 \rangle$, for all $i, 1 \leq i \leq m$ there exists a $\gamma \in \Delta^{\mathcal{I}}$ with $r^{\mathcal{I}}(f(\mu), \gamma) = 1$, and as $\mathcal{I}$ satisfies all the axioms of the form $\langle C \overset{r}{\rightsquigarrow} D \rangle \in \mathcal{T}_{\mathcal{P}}$, it follows that

$$A^{\mathcal{I}}(\gamma) = 2^{-v_\mu v_i} = 2^{-v_{\mu i}}, \quad B^{\mathcal{I}}(\gamma) = 2^{-w_\mu w_i} = 2^{-w_{\mu i}},$$

$H^{\mathcal{I}}(\gamma) = 0.5$ and for all $j, 1 \leq j \leq m$, $V_j^{\mathcal{I}}(\gamma) = 2^{-v_j}, W_j^{\mathcal{I}}(\gamma) = 2^{-w_j}$. Setting $f(\mu i) := \gamma$ thus satisfies the required property. □

6.5 Finding a Solution

From this lemma it follows that, if the PCP $\mathcal{P}$ has a solution $\mu \in \{1,\dots,m\}^*$, then every witnessed model $\mathcal{I}$ of $\mathcal{A}_\mathcal{P}$ and $\mathcal{T}_\mathcal{P}$ contains a node $\delta = f(\mu)$ such that $A^\mathcal{I}(\delta) = B^\mathcal{I}(\delta)$; that is, where A and B encode the same word. Conversely, if every witnessed model contains such a node, then in particular $\mathcal{I}_\mathcal{P}$ does, and thus $\mathcal{P}$ has a solution. The question is how to detect whether a node with this characteristics exists in every model. We will extend $\mathcal{T}_\mathcal{P}$ with axioms that further restrict $\mathcal{I}_\mathcal{P}$ to satisfy $A^{\mathcal{I}_\mathcal{P}}(\mu) \neq B^{\mathcal{I}_\mathcal{P}}(\mu)$ for every $\mu \in \{1,\dots,m\}^*$. This will ensure that the extended ontology will have a model iff $\mathcal{P}$ has no solution.

Suppose for now that, for some $\mu \in \{1,\dots,m\}^*$, it holds that

$$2^{-v_\mu} = A^{\mathcal{I}_\mathcal{P}}(\mu) > B^{\mathcal{I}_\mathcal{P}}(\mu) = 2^{-w_\mu}.$$

We then have that $v_\mu < w_\mu$ and hence $w_\mu - v_\mu \geq 1$. It thus follows that

$$A^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow B^{\mathcal{I}_\mathcal{P}}(\mu) = 2^{-w_\mu}/2^{-v_\mu} = 2^{-(w_\mu - v_\mu)} \leq 2^{-1} = 0.5.$$

Likewise, if $A^{\mathcal{I}_\mathcal{P}}(\mu) < B^{\mathcal{I}_\mathcal{P}}(\mu)$, we get $B^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow A^{\mathcal{I}_\mathcal{P}}(\mu) \leq 0.5$. Additionally, if we have $A^{\mathcal{I}_\mathcal{P}}(\mu) = B^{\mathcal{I}_\mathcal{P}}(\mu)$, then it is easy to verify that

$$A^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow B^{\mathcal{I}_\mathcal{P}}(\mu) = B^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow A^{\mathcal{I}_\mathcal{P}}(\mu) = 1.$$

From all this it follows that, for every $\mu \in \{1,\dots,m\}^*$,

$$\begin{aligned} A^{\mathcal{I}_\mathcal{P}}(\mu) \neq B^{\mathcal{I}_\mathcal{P}}(\mu) \quad \text{iff} \quad & \text{either } A^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow B^{\mathcal{I}_\mathcal{P}}(\mu) \leq 0.5 \\ & \text{or } B^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow A^{\mathcal{I}_\mathcal{P}}(\mu) \leq 0.5. \end{aligned} \tag{6.1}$$

Thus, the instance $\mathcal{P}$ has no solution iff for every $\mu \in \{1,\dots,m\}^*$ one of the two restrictions $A^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow B^{\mathcal{I}_\mathcal{P}}(\mu) \leq 0.5$ and $B^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow A^{\mathcal{I}_\mathcal{P}}(\mu) \leq 0.5$ is satisfied. We now define the TBox $\mathcal{T}'_\mathcal{P}$ that ensures this behavior in every model. Let

$$\mathcal{T}'_\mathcal{P} := \mathcal{T}_\mathcal{P} \cup \{\langle X \sqcap A \sqsubseteq X \sqcap B \sqcap H \geq 1\rangle,\ \langle \neg X \sqcap B \sqsubseteq \neg X \sqcap A \sqcap H \geq 1\rangle\}.$$

We finally have the desired result.

Lemma 6.4 *The instance $\mathcal{P}$ of the PCP has a solution iff $\mathcal{A}_\mathcal{P}$ is not witnessed consistent w.r.t. $\mathcal{T}'_\mathcal{P}$.*

Proof Assume that $\mathcal{P}$ has a solution $\mu = i_1 \cdots i_k$ and let $u = v_\mu = w_\mu$. Suppose there is a witnessed model $\mathcal{I}$ of $\mathcal{A}_\mathcal{P}$ and $\mathcal{T}'_\mathcal{P}$. Since $\mathcal{T}_\mathcal{P} \subseteq \mathcal{T}'_\mathcal{P}$, $\mathcal{I}$ is also a model of $\mathcal{T}_\mathcal{P}$. From Lemma 6.3 it follows that there is a node $\delta \in \Delta^\mathcal{I}$ where $A^\mathcal{I}(\delta) = A^{\mathcal{I}_\mathcal{P}}(\mu) = B^{\mathcal{I}_\mathcal{P}}(\mu) = B^\mathcal{I}(\delta)$. Then, $A^\mathcal{I}(\delta) \Rightarrow B^\mathcal{I}(\delta) = 1$ and $B^\mathcal{I}(\delta) \Rightarrow A^\mathcal{I}(\delta) = 1$. Since $\mathcal{I}$ is a model of $\langle X \sqsubseteq X \sqcap X \geq 1\rangle$, we have that $X^\mathcal{I}(\delta) \leq (X^\mathcal{I}(\delta))^2$, and hence $X^\mathcal{I}(\delta) \in \{0,1\}$.

If $X^\mathcal{I}(\delta) = 1$, then $(X \sqcap A)^\mathcal{I}(\delta) = A^\mathcal{I}(\delta)$ and

$$(X \sqcap B \sqcap H)^\mathcal{I}(\delta) = B^\mathcal{I}(\delta) \cdot 0.5 = A^\mathcal{I}(\delta)/2 < A^\mathcal{I}(\delta),$$

which violates the axiom $\langle X \sqcap A \sqsubseteq X \sqcap B \sqcap H \geq 1\rangle$. Alternatively, if $X^\mathcal{I}(\delta) = 0$, then the axiom $\langle \neg X \sqcap B \sqsubseteq \neg X \sqcap A \sqcap H \geq 1\rangle$ is violated. Thus $\mathcal{I}$ cannot be a model of $\mathcal{T}'_\mathcal{P}$.

For the converse, assume that $\mathcal{A}_\mathcal{P}$ is not witnessed consistent w.r.t. $\mathcal{T}'_\mathcal{P}$. Then $\mathcal{I}_\mathcal{P}$ is not a model of $\mathcal{A}_\mathcal{P}$ and $\mathcal{T}'_\mathcal{P}$. Since it is a model of $\mathcal{A}_\mathcal{P}$ and $\mathcal{T}_\mathcal{P}$, $\mathcal{I}_\mathcal{P}$ violates some axiom in $\mathcal{T}'_\mathcal{P} \setminus \mathcal{T}_\mathcal{P}$. Thus, there is a node $\mu \in \{1,\dots,m\}^*$ where $\mathcal{I}_\mathcal{P}$ violates one axiom in $\mathcal{T}'_\mathcal{P} \setminus \mathcal{T}_\mathcal{P}$. If it violates the first axiom, then $X^{\mathcal{I}_\mathcal{P}}(\mu) \cdot A^{\mathcal{I}_\mathcal{P}}(\mu) > X^{\mathcal{I}_\mathcal{P}}(\mu) \cdot B^{\mathcal{I}_\mathcal{P}}(\mu) \cdot H^{\mathcal{I}_\mathcal{P}}(\mu)$. Then

we obtain the following two consequences: (i) $X^{\mathcal{I}_\mathcal{P}}(\mu) = 1$ and $A^{\mathcal{I}_\mathcal{P}}(\mu) \leq B^{\mathcal{I}_\mathcal{P}}(\mu)$, and (ii) $A^{\mathcal{I}_\mathcal{P}}(\mu) > B^{\mathcal{I}_\mathcal{P}}(\mu)/2$. From (i) we have that $A^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow B^{\mathcal{I}_\mathcal{P}}(\mu) = 1 > 0.5$. From (ii) it follows that $B^{\mathcal{I}_\mathcal{P}}(\mu) \Rightarrow A^{\mathcal{I}_\mathcal{P}}(\mu) = A^{\mathcal{I}_\mathcal{P}}(\mu)/B^{\mathcal{I}_\mathcal{P}}(\mu) > 0.5$. Thus, $A^{\mathcal{I}_\mathcal{P}}(\mu) = B^{\mathcal{I}_\mathcal{P}}(\mu)$ (see (6.1) above) and μ is a solution of $\mathcal{P}$. Analogously, if the second axiom is violated, then $\mathcal{P}$ has a solution. □

This implies that we can reduce the PCP to ABox consistency in Π-$\mathcal{ALC}$, and in particular, that the latter problem is undecidable.

Theorem 6.5 *Witnessed consistency of Π-$\mathcal{ALC}$ ABoxes w.r.t. TBoxes is undecidable.*

6.6 Beyond Π-$\mathcal{ALC}$

Using similar techniques to those presented above, it is possible to generalize the undecidability result to fuzzy DLs using other continuous t-norms for their semantics. In fact, it has been shown in [13, 15] that for every continuous t-norm $\otimes$ that is not the Gödel t-norm, consistency in $\otimes$-$\mathcal{ALC}$ w.r.t. TBoxes is undecidable. This yields the following theorem.

Theorem 6.6 *Witnessed consistency of $\otimes$-$\mathcal{ALC}$ ABoxes w.r.t. TBoxes is undecidable if $\otimes$ is not the Gödel t-norm.*

Recall that the construction of the ABox $\mathcal{A}_\mathcal{P}$ used for proving Theorem 6.5 depends strongly on the use of equality concept assertions. More precisely, we used axioms of the form $\langle A(a) = 2^{-v_1} \rangle$, among others, in our reduction from the PCP. It can be shown that if equality concept assertions are disallowed, then the consistency problem of Π-$\mathcal{ALC}$ can be reduced in linear time to consistency in crisp $\mathcal{ALC}$, and hence can be decided in exponential time.

Definition 6.7 An ABox $\mathcal{A}$ is called *equality-free* if there are no assertions of the form $\langle C(a) = p \rangle$ or $\langle r(a,b) = p \rangle$ in $\mathcal{A}$.

Theorem 6.8 ([11]) *For the fuzzy DL Π-$\mathcal{ALC}$, witnessed consistency of equality-free ABoxes w.r.t. TBoxes is* EXPTIME*-complete.*

The linear-time reduction to crisp $\mathcal{ALC}$ depends only on having a t-norm without zero-divisors. This yields a decidability (in EXPTIME) result for the consistency problem in $\otimes$-$\mathcal{ALC}$ for any t-norm $\otimes$ that does not start with Łukasiewicz. Unfortunately, for all other t-norms the problem is still undecidable, even if restricted to equality-free ABoxes. The following theorem is a consequence of the results from [11, 13].

Theorem 6.9 *Witnessed consistency of equality-free ABoxes w.r.t. TBoxes in $\otimes$-$\mathcal{ALC}$ is decidable (in exponential time) iff $\otimes$ has no zero-divisors.*

Theorems 6.6 and 6.9 give a classification of the decidability status of consistency in fuzzy $\mathcal{ALC}$ with general concept inclusions. These results partition this family of fuzzy DLs into those that are undecidable, and those that can be reduced to crisp reasoning.

7 Conclusions

Although the tableau algorithm presented in this paper does not provide a decision procedure, we nevertheless think that it is interesting since it localizes the problem that may cause undecidability: one can repair the problem of non-termination by blocking, but the system of equations produced by a terminating run of the tableau algorithm is infinite, and thus it is not clear how to solve it even though solvability of finite systems is decidable. Theorems 6.5 and 6.6 show that this problem is actually undecidable for most t-norms. The undecidability proof of witnessed consistency in Π-$\mathcal{ALC}$ is basically the one first published in [4], though there we considered a somewhat different DL (additionally containing an implication constructor, but neither disjunction nor negation) and the whole class of t-norms "starting with product."

In this paper we have considered only consistency of ontologies. In contrast to the crisp case, other inference problems (such as subsumption and the instance problem) cannot necessarily be reduced to consistency for the fuzzy case. Thus, (un)decidability results do not necessarily transfer from consistency to these other problems. In fact, it was shown in [11] that a reduction to crisp reasoning cannot be used for deciding subsumption, even in the cases where this is possible for consistency. As future work, we plan to study the decidability status of these problems as well.

Another open question is whether the (un)decidability results still hold if the restriction to witnessed models is removed. First steps in this direction have been made, showing that consistency is undecidable for every t-norm with zero-divisors [12], and decidable for t-norms without zero-divisors if the ABox is equality-free and the constructor $\forall$ is not allowed [10]. We will try to fully characterize (un)decidability w.r.t. general models, as was done for witnessed models.

References

1. Baader, F., Bürckert, H.J., Nebel, B., Nutt, W., Smolka, G.: On the expressivity of feature logics with negation, functional uncertainty, and sort equations. Journal of Logic, Language and Information **2**, 1–18 (1993)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
3. Baader, F., Peñaloza, R.: Are fuzzy description logics with general concept inclusion axioms decidable? In: Proceedings of the 2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011), pp. 1735–1742. IEEE Press (2011)
4. Baader, F., Peñaloza, R.: On the undecidability of fuzzy description logics with GCIs and product t-norm. In: C. Tinelli, V. Sofronie-Stokkermans (eds.) Proceedings of 8th International Symposium on Frontiers of Combining Systems (FroCoS 2011), *Lecture Notes in Computer Science*, vol. 6989, pp. 55–70. Springer-Verlag (2011)
5. Baader, F., Sattler, U.: An overview of tableau algorithms for description logics. Studia Logica **69**, 5–40 (2001)
6. Bobillo, F., Bou, F., Straccia, U.: On the failure of the finite model property in some fuzzy description logics. Fuzzy Sets and Systems **172**(1), 1–12 (2011). DOI 10.1016/j.fss.2011.02.012
7. Bobillo, F., Straccia, U.: A fuzzy description logic with product t-norm. In: Proceedings of the 2007 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2007), pp. 1–6. IEEE Press (2007)
8. Bobillo, F., Straccia, U.: On qualified cardinality restrictions in fuzzy description logics under Lukasiewicz semantics. In: Proceedings of the 12th International Conference on Information Processing and Managment of Uncertainty in Knowledge-Based Systems (IPMU 2008), pp. 1008–1015 (2008)
9. Bobillo, F., Straccia, U.: Fuzzy description logics with general t-norms and datatypes. Fuzzy Sets and Systems **160**(23), 3382–3402 (2009)
10. Borgwardt, S., Distel, F., Peñaloza, R.: Gödel negation makes unwitnessed consistency crisp. In: Y. Kazakov, D. Lembo, F. Wolter (eds.) Proceedings of the 2012 International Workshop on Description Logics (DL 2012), *CEUR Workshop Proceedings*, vol. 846, pp. 103–113 (2012)

11. Borgwardt, S., Distel, F., Peñaloza, R.: How fuzzy is my fuzzy description logic? In: B. Gramlich, D. Miller, U. Sattler (eds.) Proceedings of the 6th International Joint Conference on Automated Reasoning (IJCAR 2012), *Lecture Notes in Artificial Intelligence*, vol. 7364, pp. 82–96. Springer-Verlag (2012)
12. Borgwardt, S., Peñaloza, R.: Non-Gödel negation makes unwitnessed consistency undecidable. In: Y. Kazakov, D. Lembo, F. Wolter (eds.) Proceedings of the 2012 International Workshop on Description Logics (DL 2012), *CEUR Workshop Proceedings*, vol. 846, pp. 411–421 (2012). Poster paper.
13. Borgwardt, S., Peñaloza, R.: Undecidability of fuzzy description logics. In: G. Brewka, T. Eiter, S. McIlraith (eds.) Proceedings of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR 2012), pp. 232–242. AAAI Press (2012)
14. Buchheit, M., Donini, F.M., Schaerf, A.: Decidable reasoning in terminological knowledge representation systems. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI 1993), pp. 704–709. Morgan Kaufmann, Los Altos (1993)
15. Cerami, M., Straccia, U.: On the (un)decidability of fuzzy description logics under Łukasiewicz t-norm. Information Sciences **227**, 1–21 (2013). DOI 10.1016/j.ins.2012.11.019
16. García-Cerdaña, A., Armengol, E., Esteva, F.: Fuzzy description logics and t-norm based fuzzy logics. International Journal of Approximate Reasoning **51**, 632–655 (2010)
17. Hájek, P.: Metamathematics of Fuzzy Logic (Trends in Logic). Springer-Verlag (2001)
18. Hájek, P.: Making fuzzy description logic more general. Fuzzy Sets and Systems **154**(1), 1–15 (2005)
19. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From SHIQ and RDF to OWL: The making of a web ontology language. Journal of Web Semantics **1**(1), 7–26 (2003)
20. Horrocks, I., Sattler, U.: A tableaux decision procedure for $\mathcal{SHOIQ}$. In: L.P. Kaelbling, A. Saffiotti (eds.) Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), pp. 448–453. Professional Book Center (2005)
21. Klement, E.P., Mesiar, R., Pap, E.: Triangular Norms. Springer-Verlag (2000)
22. Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in description logics for the semantic web. Journal of Web Semantics **6**(4), 291–308 (2008). DOI 10.1016/j.websem.2008.04.001
23. Lutz, C.: Description logics with concrete domains—a survey. In: P. Balbiani, N.Y. Suzuki, F. Wolter, M. Zakharyaschev (eds.) Advances in Modal Logics Volume 4, pp. 265–296. King's College Publications (2003)
24. McCallum, S.: Solving polynomial strict inequalities using cylindrical algebraic decomposition. The Computer Journal **36**(5), 432–438 (1993)
25. Molitor, R., Tresp, C.: Extending description logics to vague knowledge in medicine. In: P. Szczepaniak, P. Lisboa, S. Tsumoto (eds.) Fuzzy Systems in Medicine, *Studies in Fuzziness and Soft Computing*, vol. 41, pp. 617–635. Springer-Verlag (2000)
26. Mostert, P.S., Shields, A.L.: On the structure of semigroups on a compact manifold with boundary. Annals of Mathematics **65**, 117–143 (1957)
27. Post, E.: A variant of a recursively unsolvable problem. Bulletin of the American Mathematical Society **52**, 264–268 (1946)
28. Schmidt-Schauß, M., Smolka, G.: Attributive concept descriptions with complements. Artificial Intelligence **48**(1), 1–26 (1991)
29. Stoilos, G., Stamou, G.B., Tzouvaras, V., Pan, J.Z., Horrocks, I.: The fuzzy description logic f-$\mathcal{SHIN}$. In: Proceedings of the 1st International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2005), pp. 67–76 (2005)
30. Straccia, U.: Reasoning within fuzzy description logics. Journal of Artificial Intelligence Research **14**, 137–166 (2001)
31. Straccia, U.: Description logics with fuzzy concrete domains. In: Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence (UAI 2005), pp. 559–567. AUAI Press (2005)
32. Straccia, U., Bobillo, F.: Mixed integer programming, general concept inclusions and fuzzy description logics. In: Proceedings of the 5th EUSFLAT Conference, pp. 213–220. Universitas Ostraviensis (2007)
33. Tresp, C.B., Molitor, R.: A description logic for vague knowledge. In: Proceedings of the 13th European Conference on Artificial Intelligence (ECAI 1998), pp. 361–365. J. Wiley and Sons (1998)
34. Zadeh, L.A.: Fuzzy sets. Information and Control **8**(3), 338–353 (1965)