Finite Lattices Do Not Make Reasoning In $\mathcal{ALCI}$ Harder

Stefan Borgwardt and Rafael Peñaloza
{stefborg,penaloza}@tcs.inf.tu-dresden.de

Theoretical Computer Science, TU Dresden, Germany

Abstract. We consider the fuzzy logic $\mathcal{ALCI}$ with semantics based on a finite residuated lattice. We show that the problems of satisfiability and subsumption of concepts in this logic are ExpTime-complete w.r.t. general TBoxes and PSpace-complete w.r.t. acyclic TBoxes. This matches the known complexity bounds for reasoning in crisp $\mathcal{ALCI}$.

1 Introduction

OWL 2, the current standard ontology language for the semantic web, is based on the crisp description logic (DL) $\mathcal{SROIQ}(D)$. As a crisp logic, it is not well suited to express vague or imprecise concepts, such as HighTemperature, that can be found in numerous domains; prominently, in the biomedical area.

Fuzzy extensions of DLs have been studied for over a decade, and the literature on the topic is very extensive (see [15] for a survey). However, most of those approaches are based on the very simple *Zadeh semantics* where conjunction is interpreted as the minimum, with truth values ranging over the interval $[0, 1]$ of rational numbers. The last lustrum has seen a shift towards more general semantics for treating vagueness. On the one hand, the use of continuous t-norms as the underlying interpretation function for conjunction was proposed in [14]. On the other hand, [18] allows lattice-based truth values, but still restricts to Zadeh-like semantics.

Most of the work since then has focused on t-norm-based semantics over the unit interval; yet, ontologies are usually restricted to be unfoldable or acyclic [4–6]. Indeed, very recently it has been shown that general concept inclusion axioms (GCIs) can cause undecidability even in fuzzy DLs based on $\mathcal{ALC}$ [2, 3, 9, 11]. These results motivate restricting the logics, e.g. to finitely-valued semantics.

If one considers the Łukasiewicz t-norm over finitely many values, then reasoning is decidable even for very expressive DLs, as shown in [7] through a reduction to crisp reasoning. When restricted to $\mathcal{ALC}$ without terminological axioms, concept satisfiability is PSpace-complete as in the crisp case [10].[1] In the presence of general TBoxes, this problem becomes ExpTime-complete [8, 9], again matching the complexity of the crisp case, even if arbitrary (finite) lattices and t-norms are allowed. However, the complexity of subsumption of concepts

[1] The paper [10] considers a syntactic variant of fuzzy $\mathcal{ALC}$ with only one role.

was left as an open problem, as the standard reduction used in crisp DLs does not work with general t-norm semantics.

In this paper, we improve these complexity results to the fuzzy logic $\mathcal{ALCI}_L$ over finite lattices with general and acyclic TBoxes. More precisely, we show that in this logic, concept satisfiability is ExpTime-complete w.r.t. general TBoxes, and PSpace-complete w.r.t. acyclic TBoxes. Moreover, the same complexity bounds also hold for deciding subsumption between concepts.

2 Preliminaries

We will first give a short introduction to residuated lattices, which will be used for defining the semantics of our logic.[2] Afterwards, we recall some results from automata theory that will allow us to obtain tight upper bounds for the complexity of deciding satisfiability and subsumption of concepts.

2.1 Residuated Lattices

A *lattice* is an algebraic structure $(L, \vee, \wedge)$ over a *carrier set* L with two binary operations *join* $\vee$ and *meet* $\wedge$ that are idempotent, associative, and commutative and satisfy the absorption laws $\ell_1 \vee (\ell_1 \wedge \ell_2) = \ell_1 = \ell_1 \wedge (\ell_1 \vee \ell_2)$ for all $\ell_1, \ell_2 \in L$. L induces the ordering $\ell_1 \leq \ell_2$ iff $\ell_1 \wedge \ell_2 = \ell_1$ for all $\ell_1, \ell_2 \in L$. L is called *distributive* if $\vee$ and $\wedge$ distribute over each other, *finite* if L is finite, and *bounded* if it has a *minimum* and a *maximum* element, denoted as $\mathbf{0}$ and $\mathbf{1}$, respectively. It is *complete* if joins and meets of arbitrary subsets $T \subseteq L$, denoted by $\bigvee_{t\in T} t$ and $\bigwedge_{t \in T} t$ respectively, exist. Every finite lattice is also bounded and complete. Whenever it is clear from the context, we will simply use the carrier set L to represent the lattice $(L, \vee, \wedge)$.

A *De Morgan lattice* is a bounded distributive lattice extended with an involutive and anti-monotonic unary operation $\sim$, called *(De Morgan) negation*, satisfying the De Morgan laws $\sim(\ell_1 \vee \ell_2) = \sim\ell_1 \wedge \sim\ell_2$ and $\sim(\ell_1 \wedge \ell_2) = \sim\ell_1 \vee \sim\ell_2$ for all $\ell_1, \ell_2 \in L$.

A *residuated lattice* is a lattice L extended with two binary operators $\otimes$ (called *t-norm*) and $\Rightarrow$ (called *residuum*) such that $\otimes$ is associative, commutative, and has $\mathbf{1}$ as its unit and for every $\ell_1, \ell_2, \ell_3 \in L$, $\ell_1 \otimes \ell_2 \leq \ell_3$ iff $\ell_2 \leq \ell_1 \Rightarrow \ell_3$ holds. In a complete residuated lattice L, $\ell_1 \Rightarrow \ell_2 = \bigvee\{x \mid \ell_1 \otimes x \leq \ell_2\}$.[3] A simple consequence of this is that for every $\ell_1, \ell_2 \in L$, (i) $\mathbf{1} \Rightarrow \ell_1 = \ell_1$, and (ii) $\ell_2 \leq \ell_2$ iff $\ell_1 \Rightarrow \ell_2 = \mathbf{1}$. Additionally, the t-norm $\otimes$ is always monotonic.

In a residuated De Morgan lattice L, one can define the *t-conorm* $\oplus$ as $\ell_1 \oplus \ell_2 := \sim(\sim\ell_1 \otimes \sim\ell_2)$. For example, the meet operator $\ell_1 \wedge \ell_2$ defines a t-norm; its t-conorm is $\ell_1 \vee \ell_2$.

[2] For a more comprehensive view on residuated lattices, we refer the reader to [13, 12].

[3] We could also define the operator $\Rightarrow$ using this supremum, even if the complete lattice L is not residuated without affecting the results from Section 4.

In the following section, we will describe the fuzzy description logic $\mathcal{ALCI}_L$, whose semantics uses the residuum $\Rightarrow$ and the negation $\sim$. We emphasize, however, that the reasoning algorithm presented in Section 4 can be used with any choice of operators, as long as these are computable. In particular this means that our algorithm could also deal with other variants of fuzzy semantics, e.g. so-called Zadeh semantics [8, 18].

2.2 PSpace Automata

To obtain upper bounds for the complexity of reasoning in $\mathcal{ALCI}_L$, we will make a reduction to the emptiness problem of looping automata on infinite trees. These automata receive as input the (unlabeled) infinite k-ary tree K^* for $K := \{1, \ldots, k\}$ with $k \in \mathbb{N}$. The *nodes* of this tree are represented as words in K^*: the empty word ε represents the root node, and ui represents the i-th successor of the node u. A *path* is a sequence $v_1, \ldots, v_m$ of nodes such that $v_1 = \varepsilon$ and each v_{i+1} is a direct successor of v_i.

Definition 1 (looping automaton). *A* looping automaton (LA) *is a tuple $\mathcal{A} = (Q, I, \Delta)$ where Q is a finite set of* states*, $I \subseteq Q$ a set of* initial states*, and $\Delta \subseteq Q \times Q^k$ the* transition relation*. A* run *of $\mathcal{A}$ is a mapping $r : K^* \to Q$ assigning states to each node of K^* such that $r(\varepsilon) \in I$ and for every $u \in K^*$ we have $(r(u), r(u1), \ldots, r(uk)) \in \Delta$. The* emptiness problem *for LA is to decide whether a given LA has a run.*

The emptiness of LA can be decided in polynomial time using a bottom-up approach [19]. Alternatively, one can use a top-down approach, which relies on the fact that if there is a run, then there is also a periodic run. To speed up the top-down search, one wants to find the period of a run as early as possible. This motivates the notion of *blocking automata*.

Definition 2 (m-blocking). *Let $\mathcal{A} = (Q, \Delta, I)$ be a looping automaton. We say that $\mathcal{A}$ is m-*blocking *for $m \in \mathbb{N}$ if every path $v_1, \ldots, v_m$ of length m in a run r of $\mathcal{A}$ contains two nodes v_i and v_j $(i < j)$ such that $r(v_j) = r(v_i)$.*

Clearly, every looping automaton is m-blocking for every $m > |Q|$. However, the main interest in blocking automata arises when one can find a smaller bound on m. One way to reduce this limit is through a so-called *faithful* family of functions.

Definition 3 (faithful). *Let $\mathcal{A} = (Q, \Delta, I)$ be a looping automaton on k-ary trees. The family of functions $f_q : Q \to Q$ for $q \in Q$ is* faithful *w.r.t. $\mathcal{A}$ if for all $q, q_0, q_1, \ldots, q_k \in Q$,*

- *if $(q, q_1, \ldots, q_k) \in \Delta$, then $(q, f_q(q_1), \ldots, f_q(q_k)) \in \Delta$, and*
- *if $(q_0, q_1, \ldots, q_k) \in \Delta$, then $(f_q(q_0), f_q(q_1), \ldots, f_q(q_k)) \in \Delta$.*

The subautomaton *$\mathcal{A}^S = (Q, \Delta^S, I)$ of $\mathcal{A}$ induced by this family has the transition relation $\Delta^S = \{(q, f_q(q_1), \ldots, f_q(q_k)) \mid (q, q_1, \ldots, q_k) \in \Delta\}$.*

Lemma 4 ([1]). *Let $\mathcal{A}$ be a looping automaton and $\mathcal{A}^S$ its subautomaton induced by a faithful family of functions. $\mathcal{A}$ has a run iff $\mathcal{A}^S$ has a run.*

The construction that we will present in Section 4 produces automata that are exponential on the size of the input. For such cases, it has been shown that if the automata are m-blocking for some m bounded polynomially on the size of the input (that is, logarithmically on the size of the automaton), then the emptiness test requires only polynomial space.

Definition 5 (PSpace **on-the-fly construction).** *Assume that we have a set $\mathfrak{I}$ of inputs and a construction that yields, for every $\mathfrak{i} \in \mathfrak{I}$, an $m_\mathfrak{i}$-blocking automaton $\mathcal{A}_\mathfrak{i} = (Q_\mathfrak{i}, \Delta_\mathfrak{i}, I_\mathfrak{i})$ working on $k_\mathfrak{i}$-ary trees. This construction is called a* PSpace on-the-fly construction *if there is a polynomial P such that, for every input $\mathfrak{i}$ of size n*

- *$m_\mathfrak{i} \leq P(n)$ and $k_\mathfrak{i} \leq P(n)$,*
- *every element of $Q_\mathfrak{i}$ is of a size bounded by $P(n)$, and*
- *one can non-deterministically guess in time bounded by $P(n)$ an element of $I_\mathfrak{i}$, and, for a state $q \in Q_\mathfrak{i}$, a transition from $\Delta_\mathfrak{i}$ with first component q.*

Theorem 6 ([1]). *If the looping automata $\mathcal{A}_\mathfrak{i}$ are obtained from the inputs $\mathfrak{i} \in \mathfrak{I}$ by a* PSpace *on-the-fly construction, then the emptiness problem for $\mathcal{A}_\mathfrak{i}$ can be decided in* PSpace.

In Section 5 we will use this theorem to give PSpace upper bounds on the complexity of reasoning in the logic $\mathcal{ALCI}_L$, which we introduce next.

3 The Fuzzy Logic $\mathcal{ALCI}_L$

For the rest of this paper, L denotes a fixed residuated, complete De Morgan lattice with the t-norm $\otimes$. The fuzzy description logic $\mathcal{ALCI}_L$ is a generalization of the crisp DL $\mathcal{ALCI}$ that uses the elements of L as truth values, instead of just the Boolean *true* and *false*. The syntax of $\mathcal{ALCI}_L$ is the same as in $\mathcal{ALCI}$: given the sets $\mathsf{N_C}$ and $\mathsf{N_R}$ of concept and role names, the set of *complex roles* is $\mathsf{N_R} \cup \{r^- \mid r \in \mathsf{N_R}\}$, and $\mathcal{ALCI}_L$ concepts are built using the syntactic rule

$$C ::= A \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \neg C \mid \exists s.C \mid \forall s.C \mid \top \mid \bot,$$

where $A \in \mathsf{N_C}$ and s is a complex role. For a complex role s, the *inverse of s* (denoted by $\overline{s}$) is s^- if $s \in \mathsf{N_R}$ and r if $s = r^-$.

The semantics of this logic is based on interpretation functions that map every concept C to a function specifying the membership degree of every domain element to C.

Definition 7 (semantics of $\mathcal{ALCI}_L$). *An* interpretation *is a pair $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ where $\Delta^\mathcal{I}$ is a non-empty (crisp)* domain *and $\cdot^\mathcal{I}$ is a function that assigns to every concept name A and every role name r functions $A^\mathcal{I} : \Delta^\mathcal{I} \to L$ and $r^\mathcal{I} : \Delta^\mathcal{I} \times \Delta^\mathcal{I} \to L$, respectively. The function $\cdot^\mathcal{I}$ is extended to $\mathcal{ALCI}_L$ concepts as follows for every $x \in \Delta^\mathcal{I}$:*

- $\top^{\mathcal{I}}(x) = \mathbf{1}, \ \bot^{\mathcal{I}}(x) = \mathbf{0}$,
- $(C \sqcap D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \otimes D^{\mathcal{I}}(x), \ \ (C \sqcup D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \oplus D^{\mathcal{I}}(x)$,
- $(\neg C)^{\mathcal{I}}(x) = {\sim} C^{\mathcal{I}}(x)$,
- $(\exists s.C)^{\mathcal{I}}(x) = \bigvee_{y \in \Delta^{\mathcal{I}}} s^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y)$,
- $(\forall s.C)^{\mathcal{I}}(x) = \bigwedge_{y \in \Delta^{\mathcal{I}}} s^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y)$,

where $(r^-)^{\mathcal{I}}(x,y) = r^{\mathcal{I}}(y,x)$ *for all* $x, y \in \Delta^{\mathcal{I}}$ *and* $r \in \mathsf{N_R}$.

Notice that, unlike in crisp $\mathcal{ALCI}$, existential and universal quantifiers are not dual to each other, i.e. in general, $(\neg\exists s.C)^{\mathcal{I}}(x) = (\forall s.\neg C)^{\mathcal{I}}(x)$ does not hold.

The axioms of this logic also have an associated lattice value, which expresses the degree to which the restriction must be satisfied.

Definition 8 (axioms). Terminological axioms *are* (labeled) concept definitions *of the form* $\langle A \doteq C, \ell\rangle$ *or* (labeled) general concept inclusions (GCIs) $\langle C \sqsubseteq D, \ell\rangle$*, where* $A \in \mathsf{N_C}$*,* C, D *are* $\mathcal{ALCI}_L$ *concepts, and* $\ell \in L$.

A general TBox *is a finite set of GCIs. An* acyclic TBox *is a finite set of concept definitions such that every concept name occurs at most once in the left-hand side of an axiom, and there is no cyclic dependency between definitions. A* TBox *is either a general TBox or an acyclic TBox.*[4]

An interpretation $\mathcal{I}$ satisfies *the concept definition* $\langle A \doteq C, \ell\rangle$ *if for every* $x \in \Delta^{\mathcal{I}}$, $(A^{\mathcal{I}}(x) \Rightarrow C^{\mathcal{I}}(x)) \otimes (C^{\mathcal{I}}(x) \Rightarrow A^{\mathcal{I}}(x)) \geq \ell$ *holds. It satisfies the GCI* $\langle C \sqsubseteq D, \ell\rangle$ *if for every* $x \in \Delta^{\mathcal{I}}$, $C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x) \geq \ell$. $\mathcal{I}$ *is a* model *of the TBox* $\mathcal{T}$ *if it satisfies all axioms in* $\mathcal{T}$.

If $\mathcal{T}$ is an acyclic TBox, then all concept names occuring on the left-hand side of some axiom of $\mathcal{T}$ are called *defined*, all others are called *primitive*. If $\mathcal{T}$ is a general TBox, then all concept names appearing in it are *primitive*. A concept is an *atom* if it is either a primitive concept name, or it is a quantified concept, i.e. a concept of the form $\exists s.C$ or $\forall s.C$ for some complex role s and concept C.

We emphasize here that $\mathcal{ALCI}$ is a special case of $\mathcal{ALCI}_L$, where the underlying lattice contains only the elements $\mathbf{0}$ and $\mathbf{1}$, which may be interpreted as *false* and *true*, respectively, and the t-norm and t-conorm are just conjunction and disjunction, respectively. Accordingly, one can generalize the reasoning problems for $\mathcal{ALCI}$ to the use of other lattices. We will focus on deciding strong ℓ-satisfiability and ℓ-subsumption [8].

Definition 9 (satisfiability, subsumption). *Let* C, D *be* $\mathcal{ALCI}_L$ *concepts,* $\mathcal{T}$ *a TBox, and* $\ell \in L$. C *is* strongly ℓ-satisfiable *w.r.t.* $\mathcal{T}$ *if there is a model* $\mathcal{I}$ *of* $\mathcal{T}$ *and an* $x \in \Delta^{\mathcal{I}}$ *such that* $C^{\mathcal{I}}(x) \geq \ell$. C *is* ℓ-subsumed *by* D *w.r.t.* $\mathcal{T}$ *if every model* $\mathcal{I}$ *of* $\mathcal{T}$ *is also a model of* $\langle C \sqsubseteq D, \ell\rangle$.

In previous work we have shown that satisfiability is undecidable in $\mathcal{ALC}_L$ [9], and hence also in $\mathcal{ALCI}_L$, in general. For this reason, we assume that L is

[4] Notice that we do not consider mixed TBoxes. We could allow axioms of the form $\langle A \sqsubseteq C, \ell\rangle$ in acyclic TBoxes, as long as they do not introduce cyclic dependencies. To avoid overloading the notation, we exclude this case.

finite for the rest of this paper. As we will show in the next sections, under this restriction we obtain the same complexity upper bounds for deciding satisfiability and subsumption as in the crisp case; that is, the lattice based semantics do not increase the complexity of the logic.

4 Deciding Strong Satisfiability and Subsumption

Recall that the semantics of the quantifiers require the computation of a supremum or infimum of the membership degrees of a possibly infinite set of elements of the domain. To obtain an effective decision procedure, one usually restricts reasoning to witnessed models [14].

Definition 10 (witnessed model). *Let $n \in \mathbb{N}$. A model $\mathcal{I}$ of a TBox $\mathcal{T}$ is called* n-witnessed *if for every $x \in \Delta^{\mathcal{I}}$ and every concept of the form $\exists r.C$ there are n elements $x_1, \dots, x_n \in \Delta^{\mathcal{I}}$ such that*

$$(\exists r.C)^{\mathcal{I}}(x) = \bigvee_{i=1}^{n} r^{\mathcal{I}}(x, x_i) \otimes C^{\mathcal{I}}(x_i),$$

and analogously for the universal restrictions $\forall r.C$. In particular, if $n = 1$, then the suprema and infima from the semantics of $\exists r.C$ and $\forall r.C$ become maxima and minima, respectively. In this case, we simply say that $\mathcal{I}$ is witnessed.

We can restrict reasoning to n-witnessed models w.l.o.g.: since L is finite, we always have the n-witnessed model property for some $n \in \mathbb{N}$.

Lemma 11. *If the cardinality of the largest antichain of L is n, then $\mathcal{ALCI}_L$ has the n-witnessed model property.*

To simplify the description of the algorithm, in the following we consider $n = 1$. The algorithm and the proofs of correctness can easily be adapted for any other $n \in \mathbb{N}$.

Our algorithm for deciding satisfiability and subsumption of concepts exploits the fact that a TBox $\mathcal{T}$ has a model iff it has a well-structured tree model, called a *Hintikka tree*. Intuitively, Hintikka trees are abstract representations of models that explicitly express the membership value of all "relevant" concepts. We will construct automata that have exactly these Hintikka trees as their runs, and use the initial states to verify that an element in the model verifies the satisfiability or violates the subsumption condition, respectively. Reasoning is hence reduced to the emptiness test of these automata.

We denote by $\mathsf{sub}(C, \mathcal{T})$ the set of all subconcepts of C and of the concepts A, E, and F for all axioms $\langle E \sqsubseteq F, \ell\rangle$ or $\langle A \doteq F, \ell\rangle$ in $\mathcal{T}$. The nodes of the Hintikka trees are labeled with so-called Hintikka functions over the domain $\mathsf{sub}(C, \mathcal{T}) \cup \{\rho\}$, where ρ is an arbitrary new element, which will be used to express the degree with which the role relation to the parent node holds.

Definition 12 (Hintikka function). *A* Hintikka function *for $C, \mathcal{T}$ is a partial function $H : \mathsf{sub}(C, \mathcal{T}) \cup \{\rho\} \to L$ such that:*

(i) *H is defined for ρ and for all atoms,*
(ii) *if $H(D \sqcap E)$ is defined, then $H(D)$ and $H(E)$ are also defined and it holds that $H(D \sqcap E) = H(D) \otimes H(E)$,*
(iii) *if $H(D \sqcup E)$ is defined, then $H(D)$ and $H(E)$ are also defined and it holds that $H(D \sqcup E) = H(D) \oplus H(E)$,*
(iv) *if $H(\neg D)$ is defined, then $H(D)$ is defined and $H(\neg D) = \sim H(D)$.*

It is compatible *with the concept definition $\langle A \doteq E, \ell\rangle$ if, whenever $H(A)$ is defined, then $H(E)$ is defined and $(H(A) \Rightarrow H(E)) \otimes (H(E) \Rightarrow H(A)) \geq \ell$.*[5] *It is* compatible *with the GCI $\langle E \sqsubseteq F, \ell\rangle$ if $H(E)$ and $H(F)$ are always defined and $H(E) \Rightarrow H(F) \geq \ell$ holds.*

The Hintikka trees have a fixed arity k determined by the number of existential and universal restrictions, i.e. concepts of the form $\exists s.F$ or $\forall s.F$, contained in $\mathsf{sub}(C, \mathcal{T})$. Intuitively, each successor will act as the witness for one of these restrictions. Since we need to know which successor in the tree corresponds to which restriction, we fix an arbitrary bijection

$$\varphi : \{E \mid E \in \mathsf{sub}(C, \mathcal{T}) \text{ is of the form } \exists s.F \text{ or } \forall s.F\} \to K.$$

Definition 13 (Hintikka condition). *The tuple $(H_0, H_1, \dots, H_k)$ of Hintikka functions for $C, \mathcal{T}$ satisfies the* Hintikka condition *if:*

(i) *For every existential restriction $\exists s.G \in \mathsf{sub}(C, \mathcal{T})$*
  - *$H_{\varphi(\exists s.G)}(G)$ is defined and $H_0(\exists s.G) = H_{\varphi(\exists s.G)}(\rho) \otimes H_{\varphi(\exists s.G)}(G)$, and*
  - *$H_{\varphi(E)}(G)$ is defined and $H_0(\exists s.G) \geq H_{\varphi(E)}(\rho) \otimes H_{\varphi(E)}(G)$ for every restriction $E \in \mathsf{sub}(C, \mathcal{T})$ of the form $\exists s.F$ or $\forall s.F$.*

(ii) *For every universal restriction $\forall s.G \in \mathsf{sub}(C, \mathcal{T})$*
  - *$H_{\varphi(\forall s.G)}(G)$ is defined and $H_0(\forall s.G) = H_{\varphi(\forall s.G)}(\rho) \Rightarrow H_{\varphi(\forall s.G)}(G)$,*
  - *$H_{\varphi(E)}(G)$ is defined and $H_0(\forall s.G) \leq H_{\varphi(E)}(\rho) \Rightarrow H_{\varphi(E)}(G)$ for every restriction $E \in \mathsf{sub}(C, \mathcal{T})$ of the form $\exists s.F$ or $\forall s.F$.*

(iii) *For every existential restriction $\exists s.G \in \mathsf{sub}(C, \mathcal{T})$ and every restriction $E \in \mathsf{sub}(C, \mathcal{T})$ of the form $\exists \overline{s}.F$ or $\forall \overline{s}.F$, $H_0(G)$ is defined and $H_{\varphi(E)}(\exists s.G) \geq H_{\varphi(E)}(\rho) \otimes H_0(G)$.*

(iv) *For every universal restriction $\forall s.G \in \mathsf{sub}(C, \mathcal{T})$ and every restriction $E \in \mathsf{sub}(C, \mathcal{T})$ of the form $\exists \overline{s}.F$ or $\forall \overline{s}.F$, $H_0(G)$ is defined and $H_{\varphi(E)}(\forall s.G) \leq H_{\varphi(E)}(\rho) \Rightarrow H_0(G)$.*

The tuple is compatible *with the axiom t if the Hintikka functions $H_0, \dots, H_k$ are compatible with t.*

Condition (i) makes sure that an existential restriction $\exists s.G$ is witnessed by its designated successor $\varphi(\exists s.G)$ and all other s-successors do not contradict the witness. Condition (iii) deals with inverse roles, ensuring that the $\overline{s}$-restrictions are propagated backwards through the s-relation. Conditions (ii) and (iv) treat the universal restrictions analogously.

[5] This method, called *lazy unfolding*, is only correct for acyclic TBoxes.

A *Hintikka tree* for $C, \mathcal{T}$ is an infinite k-ary tree $\mathbf{T}$ labeled with compatible Hintikka functions for $C, \mathcal{T}$ such that $\mathbf{T}(\varepsilon)(C)$ is defined and the tuple $(\mathbf{T}(u), \mathbf{T}(u1), \dots, \mathbf{T}(uk))$ satisfies the Hintikka condition for every node $u \in K^*$. The definition of compatibility ensures that all axioms are satisfied at any node of the Hintikka tree, while the Hintikka condition makes sure that the tree is in fact a witnessed model.

The proof of the following theorem uses arguments similar to those in [1]. The main difference is the presence of successors witnessing the universal restrictions.

Theorem 14. *Let C be an $\mathcal{ALCI}_L$ concept, $\mathcal{T}$ a TBox, and $\ell \in L$. Then C is strongly ℓ-satisfiable w.r.t. $\mathcal{T}$ (in a witnessed model) iff there is a Hintikka tree $\mathbf{T}$ for $C, \mathcal{T}$ such that $\mathbf{T}(\varepsilon)(C) \geq \ell$.*

Proof (Sketch). Every witnessed model $\mathcal{I}$ of $\mathcal{T}$ with a domain element $x \in \Delta^{\mathcal{I}}$ for which $C^{\mathcal{I}}(x) \geq \ell$ holds can be *unraveled* into a Hintikka tree $\mathbf{T}$ for $C, \mathcal{T}$ as follows. We start by labeling the root node by the (total) Hintikka function that records the membership values of x for each concept from $\mathsf{sub}(C, \mathcal{T})$. We then create successors of the root by considering every $E \in \mathsf{sub}(C, \mathcal{T})$ of the form $\exists s.F$ or $\forall s.F$ and finding the witness $y \in \Delta^{\mathcal{I}}$ for this restriction. We create a new node for y which is the $\varphi(E)$-th successor of the root node and is labeled by a Hintikka set H with $H(\rho) = s^{\mathcal{I}}(x, y)$. The fact that $\mathcal{I}$ is a model of $\mathcal{T}$ ensures that these successors satisfy the Hintikka condition. By continuing this process, we construct a Hintikka tree $\mathbf{T}$ for $C, \mathcal{T}$ for which $\mathbf{T}(\varepsilon)(C) \geq \ell$ holds.

Conversely, we show that a Hintikka tree can be seen as a witnessed model with domain K^* and interpretation function given by the Hintikka functions. Notice that from the partial function labeling each node we can obtain a valuation for each concept name that satisfies all the axioms in $\mathcal{T}$. Indeed, if $\mathcal{T}$ is a general TBox, then every concept name is primitive, and hence the valuation is already defined. The fact that the Hintikka function is compatible with all the axioms in $\mathcal{T}$ implies that every node satisfies the TBox. On the other hand, if $\mathcal{T}$ is an acyclic TBox, and H is undefined for some concept names, then consider an axiom $\langle A \doteq C, \ell \rangle$ for which $H(A)$ is undefined, but $H(B)$ is defined for every atom appearing in C. The acyclicity of $\mathcal{T}$ ensures that such an axiom always exists. Thus, we can compute a value for $H(C)$ that still satisfies the conditions of Definition 12. If we set $H(A) := H(C)$, then H is still compatible with $\mathcal{T}$. By an induction argument, we can define a compatible total Hintikka function, and thus a valuation for every concept name that satisfies $\mathcal{T}$.

For this valuation to be an interpretation, it only remains to be shown that the semantics of the existential and universal restrictions are satisfied. This is ensured by the Hintikka condition. The choice of the successors also ensures that the interpretation is witnessed. As explained above, it is compatible, and hence also a model of $\mathcal{T}$. Thus, if there is a Hintikka tree $\mathbf{T}$ for $C, \mathcal{T}$ with $\mathbf{T}(\varepsilon)(C) \geq \ell$, then C is strongly ℓ-satisfiable w.r.t. $\mathcal{T}$. □

Hintikka trees can also be used for deciding (non-)subsumption between $\mathcal{ALCI}_L$ concepts. The proof of the following theorem is analogous to the one of Theorem 14.

Theorem 15. *Let C, D be $\mathcal{ALCI}_L$ concepts, $\mathcal{T}$ a TBox, and $\ell \in L$. Then C is not ℓ-subsumed by D (in a witnessed model) iff there is a Hintikka tree $\mathbf{T}$ for $C \sqcap D, \mathcal{T}$ such that $\mathbf{T}(\varepsilon)(C) \Rightarrow \mathbf{T}(\varepsilon)(D) \not\geq \ell$.*[6]

Notice that this does not yield a reduction from subsumption to satisfiability, since the residuum $\Rightarrow$ cannot in general be expressed using only the t-norm, t-conorm and negation, and in Theorem 14 the value of C at the root is restricted to a value greater or equal to ℓ, while Theorem 15 negates this restriction.

From the last two theorems it follows that satisfiability and subsumption of $\mathcal{ALCI}_L$ concepts can be reduced to deciding the existence of a Hintikka tree with additional restrictions in the root. By building looping automata whose runs correspond exactly to those Hintikka trees, we reduce $\mathcal{ALCI}_L$ reasoning to the emptiness problem of these automata. For the following, we focus only on deciding satisfiability and explain the minor modifications required for deciding subsumption.

Definition 16 (Hintikka automaton). *Let C be an $\mathcal{ALCI}_L$ concept, $\mathcal{T}$ a TBox, and $\ell \in L$. The* Hintikka automaton *for $C, \mathcal{T}, \ell$ is $\mathcal{A}_{C,\mathcal{T},\ell} = (Q, I, \Delta)$, where Q is the set of all compatible Hintikka functions for $C, \mathcal{T}$, I contains all Hintikka functions H with $H(C) \geq \ell$, and Δ is the set of all $(k+1)$-tuples of Hintikka functions that satisfy the Hintikka condition.*

The runs of $\mathcal{A}_{C,\mathcal{T},\ell}$ are exactly the Hintikka trees $\mathbf{T}$ having $\mathbf{T}(\varepsilon)(C) \geq \ell$. Thus, C is strongly ℓ-satisfiable w.r.t. $\mathcal{T}$ iff $\mathcal{A}_{C,\mathcal{T},\ell}$ is not empty. To obtain an automaton deciding ℓ-subsumption between C and D, one needs only modify the set of initial states I to contain all Hintikka functions H with $H(C) \Rightarrow H(D) \not\geq \ell$. In that case, we have that C is ℓ-subsumed by D iff the automaton is empty.

The size of the automaton $\mathcal{A}_{C,\mathcal{T},\ell}$ is exponential in the input $C, \mathcal{T}$. Hence, we have an ExpTime algorithm for this logic. For general TBoxes, this gives a tight upper bound for the complexity of satisfiability and subsumption, since these problems are already ExpTime-hard for crisp $\mathcal{ALC}$ [16].

Theorem 17. *Deciding strong satisfiability and subsumption in $\mathcal{ALCI}_L$ w.r.t. general TBoxes is* ExpTime*-complete.*

5 PSpace Results for Acyclic TBoxes

If one restricts to acyclic TBoxes, then the upper bound obtained by the emptiness test of the automaton from Definition 16 does not match the PSpace lower bound given by crisp $\mathcal{ALCI}$ with acyclic TBoxes. We will now improve this upper bound and show that satisfiability and subsumption of $\mathcal{ALCI}_L$ concepts w.r.t. acyclic TBoxes are also PSpace-complete problems.

The idea is to modify the construction of the Hintikka automata into a PSpace on-the-fly construction. Notice that $\mathcal{A}_{C,\mathcal{T},\ell}$ satisfies all but one of the

[6] Using $C \sqcap D$ only ensures that $\mathbf{T}(\varepsilon)(C)$ and $\mathbf{T}(\varepsilon)(D)$ are defined, but imposes no further restriction on their values.

conditions from Definition 5: (i) the arity of the automata is given by the number of existential and universal concepts in $\mathsf{sub}(C,\mathcal{T})$; (ii) every Hintikka function has size bounded by $|\mathsf{sub}(C,\mathcal{T})|$; (iii) building a state or a transition of the automaton requires only guessing values for all concepts in $\mathsf{sub}(C,\mathcal{T})$ and then verifying that this is indeed a valid state or transition, which can be done in time polynomial in $|\mathsf{sub}(C,\mathcal{T})|$. However, it is easy to build runs of the automata constructed by this reduction where blocking occurs only after exponentially many transitions, violating the first condition of PSpace on-the-fly constructions.

We will use a faithful family of functions to obtain a reduced automaton that guarantees blocking after at most polynomially many transitions, thus obtaining the PSpace upper bound. The idea is that it suffices to consider only transitions that reduce the maximal role depth (w.r.t. $\mathcal{T}$) in the support of the states.

The *role depth w.r.t.* $\mathcal{T}$ ($\mathsf{rd}_\mathcal{T}$) of $\mathcal{ALCI}_L$ concepts is recursively defined as follows: $\mathsf{rd}_\mathcal{T}(A) = \mathsf{rd}_\mathcal{T}(\top) = \mathsf{rd}_\mathcal{T}(\bot) = 0$ for every primitive concept name A; $\mathsf{rd}_\mathcal{T}(C \sqcap D) = \mathsf{rd}_\mathcal{T}(C \sqcup D) = \max\{\mathsf{rd}_\mathcal{T}(C), \mathsf{rd}_\mathcal{T}(D)\}$; $\mathsf{rd}_\mathcal{T}(\neg C) = \mathsf{rd}_\mathcal{T}(C)$; $\mathsf{rd}_\mathcal{T}(\exists r.C) = \mathsf{rd}_\mathcal{T}(\forall r.C) = \mathsf{rd}_\mathcal{T}(C) + 1$; and $\mathsf{rd}_\mathcal{T}(A) = \mathsf{rd}_\mathcal{T}(C)$ for every definition $\langle A \doteq C, \ell\rangle \in \mathcal{T}$. For a Hintikka function H for $C, \mathcal{T}$, we denote as $\mathsf{support}(H)$ the set of all concepts in $\mathsf{sub}(C,\mathcal{T})$ such that $H(C)$ is defined and $H(C) > \mathbf{0}$. We define $\mathsf{rd}_\mathcal{T}(H)$ as the maximum $\mathsf{rd}_\mathcal{T}(D)$ such that $D \in \mathsf{support}(H)$.

Definition 18 (functions f_H). *Let H and H' be two states of $\mathcal{A}_{C,\mathcal{T},\ell}$ with $\mathsf{rd}_\mathcal{T}(H) = n$. The function $f_H(H')$ is given by:*

$$f_H(H')(D) = \begin{cases} \mathbf{0} & \text{if } D \text{ is an atom and } \mathsf{rd}_\mathcal{T}(D) \geq n \\ H'(D) & \text{if } \mathsf{rd}_\mathcal{T}(D) < n \\ \text{undefined} & \text{otherwise.} \end{cases}$$

$$f_H(H')(\rho) = \begin{cases} \mathbf{0} & \text{if } \mathsf{support}(H) = \emptyset \\ H'(\rho) & \text{otherwise.} \end{cases}$$

Since $\mathcal{T}$ is acyclic, the function $f_H(H')$ defined above is still a Hintikka function for $C, \mathcal{T}$ compatible with all the axioms in $\mathcal{T}$.

Lemma 19. *The family of mappings f_H for states H of $\mathcal{A}_{C,\mathcal{T},\ell}$ from Definition 18 is faithful w.r.t. $\mathcal{A}_{C,\mathcal{T},\ell}$.*

Proof. Let $(H, H_1, \dots, H_k)$ be a valid transition of $\mathcal{A}_{C,\mathcal{T},\ell}$. We need to show that $(H, f_H(H_1), \dots, f_H(H_k))$ is also a transition, i.e. that it satisfies the Hintikka condition. We show in detail only the proof for the restriction (i) from Definition 13, as the others can be treated analogously.

For $\exists s.G \in \mathsf{sub}(C,\mathcal{T})$, the value $H_{\varphi(E)}(G)$ is defined for all restrictions E of the form $\exists s.F$ or $\forall s.F$ in $\mathsf{sub}(C,\mathcal{T})$. If $\mathsf{rd}_\mathcal{T}(\exists s.G) > \mathsf{rd}_\mathcal{T}(H)$, then $H(\exists s.G) = \mathbf{0}$, and all the values $f_H(H_{\varphi(E)})(G)$ are $\mathbf{0}$. Thus, the inequalities are trivially satisfied. Otherwise, $\mathsf{rd}_\mathcal{T}(G) < \mathsf{rd}_\mathcal{T}(\exists s.G) \leq \mathsf{rd}_\mathcal{T}(H)$, and thus the values $H_{\varphi(E)}(G)$ are not changed by applying f_H. If the values $H_{\varphi(E)}(\rho)$ are also left unchanged, all inequalities remain satisfied. Otherwise, $H(\exists s.G) = \mathbf{0}$, and all the values $f_H(H_{\varphi(E)})(\rho)$ are $\mathbf{0}$. Thus, the inequalities are again trivially satisfied. □

By Lemma 4, $\mathcal{A}_{C,\mathcal{T},\ell}$ is empty iff the induced subautomaton $\mathcal{A}^S_{C,\mathcal{T},\ell}$ is empty.

Theorem 20. *The construction of $\mathcal{A}^S_{C,\mathcal{T},\ell}$ from an $\mathcal{ALCI}_L$ concept C, $\ell \in L$, and an acyclic TBox $\mathcal{T}$ is a* PSpace *on-the-fly construction.*

Proof. As described before, we only need to show that the automata $\mathcal{A}^S_{C,\mathcal{T},\ell}$ are m-blocking for some m bounded polynomially in $|\mathsf{sub}(C,\mathcal{T})|$. We show that this holds for $m = \max\{\mathsf{rd}_{\mathcal{T}}(D) \mid D \in \mathsf{sub}(C,\mathcal{T})\} + 2$.

By definition of $\mathcal{A}^S_{C,\mathcal{T},\ell}$, every transition decreases the maximal role depth of the support of the state. Hence, after at most $\max\{\mathsf{rd}_{\mathcal{T}}(D) \mid D \in \mathsf{sub}(C,\mathcal{T})\}$ transitions, we reach a state H, where $H(D) = \mathbf{0}$ if D is an atom and undefined otherwise, and hence, $\mathsf{support}(H) = \emptyset$. From the next transition on, all the states additionally satisfy that $H(\rho) = \mathbf{0}$. Hence, after at most m transitions, we find two states that are equal. Since $m \leq |\mathsf{sub}(C,\mathcal{T})| + 2$, $\mathcal{A}^S_{C,\mathcal{T},\ell}$ satisfies the requirements for a PSpace on-the-fly construction. $\square$

This shows that emptiness of $\mathcal{A}^S_{C,\mathcal{T},\ell}$ and hence also of $\mathcal{A}_{C,\mathcal{T},\ell}$ is in PSpace. This yields the desired PSpace upper bound for satisfiability and similar arguments can be made for subsumption. PSpace-hardness follows from PSpace-hardness of satisfiability and subsumption w.r.t. the empty TBox in $\mathcal{ALC}$ [17].

Theorem 21. *Deciding strong satisfiability and subsumption in $\mathcal{ALCI}_L$ w.r.t. acyclic TBoxes is* PSpace*-complete.*

Notice that the definitions of Hintikka functions and Hintikka trees are independent of the operators used. One could have chosen the residual negation $\ominus\ell := \ell \Rightarrow \mathbf{0}$ to interpret the constructor $\neg$, or the Kleene-Dienes implication $\ell_1 \Rightarrow \ell_2 := {\sim}\ell_1 \vee \ell_2$ instead of the residuum. The only restrictions are that the semantics must be truth functional, i.e. the value of a formula must depend only on the values of its direct subformulas, and the underlying operators must be computable. We could also use the traditional semantics for concept definitions in which $\otimes$ is replaced by the simple meet t-norm $\wedge$.

We also point out that the algorithm can be modified for reasoning w.r.t. n-witnessed models for $n > 1$. One needs only extend the arity of the Hintikka trees to account for n witnesses for each quantified formula in $\mathsf{sub}(C,\mathcal{T})$; the arity of $\mathcal{A}_{C,\mathcal{T},\ell}$ grows polynomially in n. This does not affect the complexity upper bounds from the automata, and hence Theorems 17 and 21 still hold.

6 Conclusions

We have shown that reasoning in $\mathcal{ALCI}_L$ is not harder than in the underlying crisp DL $\mathcal{ALCI}$. More precisely, strong ℓ-satisfiability and ℓ-subsumption can be decided in ExpTime for general TBoxes and in PSpace for acyclic TBoxes. This extends the complexity results from [8–10] and demonstrates that automata can show PSpace results even for fuzzy description logics, as in the crisp case [1]. This paper provides a small step towards reasoning services for fuzzy generalizations of the current standard ontology languages, like $\mathcal{SROIQ}(D)$.

In the future, we want to study the influence of additional DL constructors and axioms on the complexity of the reasoning tasks. In particular, transitive roles, which are covered by the results in [1], have not been considered in this paper. Although in the crisp case they do not increase the complexity of checking satisfiability, it is not straightforward to generalize the methods used to show this to residuated De Morgan lattices.

Satisfiability w.r.t. general TBoxes and residuated total orders has been shown to be undecidable [9], but it remains open to find subclasses of infinite lattices and t-norms for which the problem is decidable. Over the unit interval, the product and Łukasiewicz t-norms cause undecidability w.r.t. witnessed models [3, 11]; for arbitrary models decidability is unknown in these cases.

References

1. F. Baader, J. Hladik, and R. Peñaloza. Automata can show PSPACE results for description logics. *Inform. Comput.*, 206(9-10):1045–1056, 2008.
2. F. Baader and R. Peñaloza. Are fuzzy description logics with general concept inclusion axioms decidable? In *Proc. FuzzIEEE'11*, pages 1735–1742. IEEE, 2011.
3. F. Baader and R. Peñaloza. On the undecidability of fuzzy description logics with GCIs and product t-norm. In *Proc. FroCoS'11*, pages 55–70. Springer-Verlag, 2011.
4. F. Bobillo, F. Bou, and U. Straccia. On the failure of the finite model property in some fuzzy description logics. *Fuzzy Set. Syst.*, 172(1):1–12, 2011.
5. F. Bobillo and U. Straccia. A fuzzy description logic with product t-norm. In *Proc. Fuzz-IEEE'07*, pages 1–6. IEEE, 2007.
6. F. Bobillo and U. Straccia. Fuzzy description logics with general t-norms and datatypes. *Fuzzy Set. Syst.*, 160(23):3382–3402, 2009.
7. F. Bobillo and U. Straccia. Reasoning with the finitely many-valued Łukasiewicz fuzzy description logic $\mathcal{SROIQ}$. *Inf. Sci.*, 181(4):758–778, 2011.
8. S. Borgwardt and R. Peñaloza. Description logics over lattices with multi-valued ontologies. In *Proc. IJCAI'11*, pages 768–773. AAAI Press, 2011.
9. S. Borgwardt and R. Peñaloza. Fuzzy ontologies over lattices with t-norms. In *Proc. DL'11*, pages 70–80. CEUR Workshop Proceedings, 2011.
10. F. Bou, M. Cerami, and F. Esteva. Finite-valued Łukasiewicz modal logic is PSPACE-complete. In *Proc. IJCAI'11*, pages 774–779. AAAI Press, 2011.
11. M. Cerami and U. Straccia. On the undecidability of fuzzy description logics with GCIs with Łukasiewicz t-norm. 2011. `arXiv:1107.4212v3 [cs.LO]`.
12. G. De Cooman and E. E. Kerre. Order norms on bounded partially ordered sets. *J. Fuzzy Math*, 2:281–310, 1993.
13. G. Grätzer. *General Lattice Theory, Second Edition.* Birkhäuser Verlag, 2003.
14. P. Hájek. Making fuzzy description logic more general. *FSS*, 154(1):1–15, 2005.
15. T. Lukasiewicz and U. Straccia. Managing uncertainty and vagueness in description logics for the semantic web. *J. Web Semant.*, 6(4):291–308, 2008.
16. K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. IJCAI'91*, pages 466–471. Morgan Kaufmann, 1991.
17. M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artif. Intell.*, 48(1):1–26, 1991.
18. U. Straccia. Description logics over lattices. *Int. J. Unc. Fuzz.*, 14(1):1–16, 2006.
19. M. Y. Vardi and P. Wolper. Automata-theoretic techniques for modal logics of programs. *J. Comput. Syst. Sci.*, 32(2):183–221, 1986.