TECHNISCHE UNIVERSITÄT DRESDEN

Technische Universität Dresden
Institute for Theoretical Computer Science
Chair for Automata Theory

LTCS–Report

Towards Extending the Description Logic FL0 with Threshold Concepts Using Weighted Tree Automata

Oliver Fernández Gil Pavlos Marantidis

LTCS-Report 23-04

This is an extended version of an article accepted at the 36th International Workshop on Description Logics (DL 2023).

Postal Address:
Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik
TU Dresden
01062 Dresden

http://lat.inf.tu-dresden.de

Visiting Address:
Nöthnitzer Str. 46
Dresden

Towards Extending the Description Logic FL0 with Threshold Concepts Using Weighted Tree Automata

Oliver Fernández Gil[1,3] and Pavlos Marantidis[2]

[1]Theoretical Computer Science, TU Dresden, Germany
[2]Aristotle University of Thessaloniki, Thessaloniki, Greece
[3]Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden/Leipzig, Germany

Abstract

We introduce an extension of the Description Logic $\mathcal{FL}_0$ that allows us to define concepts in an approximate way. More precisely, we extend $\mathcal{FL}_0$ with a threshold concept constructor of the form $C_{\bowtie t}$ for $\bowtie \in \{\leq, <, \geq, >\}$, whose semantics is given by using a membership distance function (mdf). A membership distance function m assigns to each domain element and concept a distance value expressing how "close" is such element to being an instance of the concept. Based on this, a threshold concept $C_{\bowtie t}$ is interpreted as the set of all domain elements that have a distance s from C such that $s \bowtie t$. We provide a framework to obtain membership distance functions based on functions that compare tuples of languages, and we show how weighted looping tree automata over a semiring can be used to define membership distance functions for $\mathcal{FL}_0$ concepts.

Contents

1 Introduction

Traditional Description Logics (DLs) [BHLS17, BCM$^+$03] are based on the semantics of classical first-order logic. This is very nice, since it allows us to formally represent conceptual knowledge of an application domain in a well-understood way. However, it can also be seen as a limitation in modeling certain application domains, whose relevant notions lack a precise definition or such a definition is very difficult to determine. More precisely, the strict interpretation of concepts (formulas) in traditional DLs only tells us whether an individual belongs to a concept or not. In view of this, representing vague or imprecise knowledge within a particular DL may require concepts of a very big size or may not be possible at all.

To alleviate this, a considerable amount of research has been directed towards extending DLs with means that would allow us to model (and reason about) imprecise knowledge. Early examples of this are *fuzzy* DLs [Yen91, Str01, Háj05], extensions of DLs with *rough semantics* [SKP07, dFEL13, PZ13], and combinations of DLs with logics that can reason about metric spaces [LWZ03, STWZ07]. More recently, three different new approaches have been proposed that allow us to define concepts in an approximate way, namely, the extensions of the DL $\mathcal{ALC}$ with automata-based *prototype distance functions* [BE16] and with *weighted combinations of concepts* [GKP$^+$19, PKR$^+$19], and the family of logics $\tau\mathcal{EL}(m)$ which extend the DL $\mathcal{EL}$ with *threshold concepts* [BBF15]. The approach proposed in [BBF15] consists of two main components: **a)** a *membership degree function* m, which instead of giving a value in $\{0, 1\}$ to evaluate membership of an individual into a concept C, gives a value in $[0, 1]$; and **b)** a new

family of concept constructors that can, for example, be used to build a threshold concept $C_{\geq t}$ from an $\mathcal{EL}$ concept C and a value $t \in [0, 1]$. The semantics of $C_{\geq t}$ is then defined as the set of individuals d that belong to C with degree (obtained by applying m to d and C) at least t. The two approaches introduced in [BE16, GKP+19, PKR+19] are conceptually similar, but use different ways of defining and combining the two components described above.

In this paper, we extend the DL $\mathcal{FL}_0$[Baa96] with threshold concepts, by following the general idea applied to $\mathcal{EL}$ in [BBF15]. The resulting family of logics is called $\mathcal{FL}_0\text{at}(m)$, where the parameter m now corresponds to a *membership distance function*. The main difference to [BBF15] is that now we use the notion of distance instead of membership degrees, to measure "how close" an element is to being an instance of a concept. Obviously, a key aspect of $\mathcal{FL}_0\text{at}(m)$ is which distance function m to choose. In this work, we provide a general framework to define such distance functions for $\mathcal{FL}_0$, which reduces the definition of membership distance functions to the definition of functions comparing tuples of formal languages. Moreover, we show that such measures can be defined by using *weighted tree automata*, which not only offers a more concrete (yet diverse) form of defining membership distance functions, but also provides a machinery that allows in many cases to actually compute these functions. A distinctive feature of this framework is that it allows us to define membership distance functions that can take into account background knowledge stated by GCIs in an $\mathcal{FL}_0$ TBox. This is not the case for the approaches proposed in [BE16, GKP+19], whereas the extension of $\mathcal{EL}$ with threshold concepts has only been extended to compute membership degrees w.r.t. an acyclic $\mathcal{EL}$ TBox [BF16].

The paper is structured as follows. In the next section, we formally introduce the DL $\mathcal{FL}_0$, and recall some related technical notions that are needed in the rest of the paper. In Section 3, we introduce the family of threshold logics $\mathcal{FL}_0\text{at}(m)$. We then continue in Section 4, by describing our framework for defining membership distance functions for $\mathcal{FL}_0$. In Section 5, we explain how weighted looping tree automata can be used to define and compute membership distance functions. We finish the paper, by summarizing the contributions of the paper and giving some ideas of how to move forward.

2 The Description Logic $\mathcal{FL}_0$

We start by formally introducing the DL $\mathcal{FL}_0$. Afterwards, we recall how subsumption (equivalence) between $\mathcal{FL}_0$ concepts can be characterized using language inclusion, and show that this idea can also be applied to characterize membership of an individual in an $\mathcal{FL}_0$ concept.

2.1 Syntax and semantics

Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be finite disjoint sets of concept names and role names, respectively. The set $\mathcal{C}_{\mathcal{FL}_0}(\mathsf{N_C}, \mathsf{N_R})$ of $\mathcal{FL}_0$ concept descriptions over $\mathsf{N_C}$ and $\mathsf{N_R}$ is obtained by using the concept constructors *conjunction* ($\sqcap$), *universal restriction* ($\forall r.C$), and *top* ($\top$) in the following way:

$$C ::= \top \mid A \mid C \sqcap C \mid \forall r.C$$

where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$ and C is an $\mathcal{FL}_0$ concept description. We will often write $\mathcal{C}_{\mathcal{FL}_0}$ in place of $\mathcal{C}_{\mathcal{FL}_0}(\mathsf{N_C}, \mathsf{N_R})$, if the sets $\mathsf{N_C}$ and $\mathsf{N_R}$ are clear from the context or irrelevant.

The semantics of $\mathcal{FL}_0$ is defined in the usual way, by using standard *first-order* interpretations. 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 subsets $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ to concept names $A \in \mathsf{N_C}$ and binary relations $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ to role names $r \in \mathsf{N_R}$. The function $\cdot^{\mathcal{I}}$ is inductively extended to all $\mathcal{FL}_0$

concepts in $\mathcal{C}_{\mathcal{FL}_0}(\mathsf{N_C}, \mathsf{N_R})$ as follows:

$$\top^{\mathcal{I}} := \Delta^{\mathcal{I}}, \quad (C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}, \quad (\forall r.C)^{\mathcal{I}} := \{d \in \Delta^{\mathcal{I}} \mid \forall e \in \Delta^{\mathcal{I}} : (d,e) \in r^{\mathcal{I}} \Rightarrow e \in C^{\mathcal{I}}\}.$$

Given two $\mathcal{FL}_0$ concepts C and D, we say that C is *subsumed* by D (written as $C \sqsubseteq D$) if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all interpretations $\mathcal{I}$. These two concepts are *equivalent* (written as $C \equiv D$) if $C \sqsubseteq D$ and $D \sqsubseteq C$. In addition, C is *satisfiable* if $C^{\mathcal{I}} \neq \emptyset$ for some interpretation $\mathcal{I}$.

An $\mathcal{FL}_0$ TBox is a finite set of general concept inclusions (GCIs), which are expressions of the form $C \sqsubseteq D$ where $C, D \in \mathcal{C}_{\mathcal{FL}_0}$. We say that an interpretation $\mathcal{I}$ is a model of $\mathcal{T}$ (written as $\mathcal{I} \models \mathcal{T}$) if it satisfies all the GCIs in $\mathcal{T}$, meaning that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all $C \sqsubseteq D$ in $\mathcal{T}$. The subsumption and equivalence relations are now defined *modulo* the set of models of $\mathcal{T}$, and denoted as $\sqsubseteq_{\mathcal{T}}$ and $\equiv_{\mathcal{T}}$, respectively. The notion of satisfiable concept is also defined modulo the set of models of $\mathcal{T}$.

2.2 $\mathcal{FL}_0$ and formal languages

In $\mathcal{FL}_0$, subsumption and equivalence can be characterized via a transition to formal languages, by utilizing a certain normal form. In particular, the semantics of $\mathcal{FL}_0$ implies that value restrictions distribute over conjunction, i.e, for all $\mathcal{FL}_0$ concepts C, D and roles r it holds that $\forall r.(C \sqcap D) \equiv \forall r.C \sqcap \forall r.D$. Using this equivalence as a rewrite rule from left to right, every $\mathcal{FL}_0$ concept description can be written as a finite conjunction of terms of the form $\forall r_1.\forall r_2. \ldots \forall r_m.A$, where $m \geq 0$, $\{r_1, \ldots, r_m\} \subseteq \mathsf{N_R}$, and $A \in \mathsf{N_C}$. We can further abbreviate such a term as $\forall w.A$, where w represents the word $r_1 \ldots r_m$ over the alphabet $\mathsf{N_R}$. In case $m = 0$, w is the empty word ε, and thus $\forall \varepsilon.A$ corresponds to A. Finally, we can group together value restrictions with the same concept name, i.e., abbreviate $\forall w_1.A \sqcap \cdots \sqcap \forall w_\ell.A$ by $\forall \{w_1, \ldots, w_\ell\}.A$, where $\{w_1, \ldots, w_\ell\}$ is a finite language over $\mathsf{N_R}$. In addition, we use the convention that $\forall \emptyset.A$ corresponds to $\top$. As a result, any two $\mathcal{FL}_0$ concept descriptions C, D over $\mathsf{N_C} = \{A_1, \ldots A_n\}$ and $\mathsf{N_R}$ can be rewritten in the normal form:

$$C \equiv \forall L_1.A_1 \sqcap \cdots \sqcap \forall L_n.A_n \qquad D \equiv \forall M_1.A_1 \sqcap \cdots \sqcap \forall M_n.A_n,$$

where $L_1, \ldots, L_n, M_1, \ldots, M_n$ are finite subsets of $\mathsf{N_R}^*$. Using this language normal form (LNF), it was shown in [BN01] that $C \sqsubseteq D$ iff $M_i \subseteq L_i$ for all $i = 1, \ldots, n$. Since $C \equiv D$ iff $C \sqsubseteq D$ and $D \sqsubseteq C$, it follows that $C \equiv D$ iff $L_i = M_i$ for all $i = 1, \ldots, n$.

In [Pen15], this characterization of subsumption was extended to non-empty TBoxes, by using the notion of value restriction sets. Given a concept $C \in \mathcal{C}_{\mathcal{FL}_0}$, a concept name $A \in \mathsf{N_C}$, and an $\mathcal{FL}_0$ TBox $\mathcal{T}$, the value restriction set of C w.r.t. $\mathcal{T}$ and A is defined as the language:

$$\mathcal{L}_{\mathcal{T}}(C, A) = \{w \in \mathsf{N_R}^* \mid C \sqsubseteq_{\mathcal{T}} \forall w.A\}.$$

Using these sets, the above characterizations of subsumption and equivalence can be lifted to take into account the GCIs in a TBox as follows (see [Pen15]):

$$C \sqsubseteq_{\mathcal{T}} D \iff \mathcal{L}_{\mathcal{T}}(D, A_i) \subseteq \mathcal{L}_{\mathcal{T}}(C, A_i) \quad (i = 1, \ldots, n),$$
$$C \equiv_{\mathcal{T}} D \iff \mathcal{L}_{\mathcal{T}}(C, A_i) = \mathcal{L}_{\mathcal{T}}(D, A_i) \quad (i = 1, \ldots, n).$$

Essentially, this means that every $\mathcal{FL}_0$ concept description C can be uniquely represented by the tuple of languages

$$\mathcal{L}_{\mathcal{T}}(C) = (\mathcal{L}_{\mathcal{T}}(C, A_1), \ldots, \mathcal{L}_{\mathcal{T}}(C, A_n))$$

and every concept description equivalent to C has the same representation.

We will now demonstrate that a similar characterization can be used to describe when an individual d is an instance of a concept description C in an interpretation $\mathcal{I}$. Given an interpretation $\mathcal{I}$, $d \in \Delta^{\mathcal{I}}$ and $A \in \mathsf{N_C}$, we define the following language:

$$\mathcal{L}_{\mathcal{I}}(d, A) = \{w \in \mathsf{N_R}^* \mid d \in (\forall w.A)^{\mathcal{I}}\}.$$

Using these languages, an individual d can be represented as a tuple of languages $(\mathcal{L}_{\mathcal{I}}(d, A_1), \ldots, \mathcal{L}_{\mathcal{I}}(d, A_n))$. Moreover, membership in $\mathcal{FL}_0$ can be characterized as follows.

Theorem 1. *Given an $\mathcal{FL}_0$ TBox $\mathcal{T}$, a model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of $\mathcal{T}$, an $\mathcal{FL}_0$ concept C, and an element $d \in \Delta^{\mathcal{I}}$ we have that $d \in C^{\mathcal{I}} \iff \mathcal{L}_{\mathcal{T}}(C, A) \subseteq \mathcal{L}_{\mathcal{I}}(d, A)$ for every $A \in N_C$.*

Proof. Let $C = \forall w_1.A_1 \sqcap \cdots \sqcap \forall w_n.A_n$. For the if direction, if $\mathcal{L}_{\mathcal{T}}(C, A) \subseteq \mathcal{L}_{\mathcal{I}}(d, A)$ for every $A \in N_C$, we have the following: initially, it is obvious that

$$\begin{aligned} w_i \in \mathcal{L}_{\mathcal{T}}(C, A) \text{ for every } i = 1, \ldots, n &\implies w_i \in \mathcal{L}_{\mathcal{I}}(d, A) \text{ for every } i = 1, \ldots, n \\ &\implies d \in (\forall w_i.A_i)^{\mathcal{I}} \text{ for every } i = 1, \ldots, n \\ &\implies d \in (\forall w_1.A_1 \sqcap \cdots \sqcap \forall w_n.A_n)^{\mathcal{I}} \end{aligned}$$

For the only if direction, assume that $d \in C^{\mathcal{I}}$. Then

$$\begin{array}{lll} w \in \mathcal{L}_{\mathcal{T}}(C, A) & \implies C \sqsubseteq_{\mathcal{T}} \forall w.A & \implies C^{\mathcal{I}} \subseteq (\forall w.A)^{\mathcal{I}} \\ & \implies d \in (\forall w.A)^{\mathcal{I}} & \implies w \in \mathcal{L}_{\mathcal{I}}(d, A) \end{array}$$

□

3 Extending $\mathcal{FL}_0$ with threshold concepts

In this section, we introduce the family of threshold logics $\mathcal{FL}_0 at(m)$.

3.1 The family of logics $\mathcal{FL}_0 at(m)$

Threshold concepts for $\mathcal{FL}_0$ are expressions of the form $C_{\bowtie s}$ where C is an $\mathcal{FL}_0$ concept, $\bowtie \in \{<, \leq, >, \geq\}$, and s is an element of a linearly ordered set $(S, \leq)$ with minimum $\mathfrak{m}$. The purpose of these concept constructors is to define sets of individuals of an interpretation $\mathcal{I}$ that may not belong to $C^{\mathcal{I}}$, but still satisfy some of the properties required by C. To quantify the notion of partial satisfaction, we use a value from S that expresses "how far" an individual d is from belonging to $C^{\mathcal{I}}$, where the $\mathfrak{m}$ identifies *crisp* membership, i.e., that $d \in C^{\mathcal{I}}$. For instance, if we consider $(S, \leq)$ as the interval $[0, 1]$ with the usual order, one can use the threshold concept $C_{\leq 0.2}$ to capture all individuals that have distance at most 0.2 from belonging to $C^{\mathcal{I}}$.

To provide the semantics for threshold concepts, we introduce the notion of membership distance function. Such a function operates as follows: given an interpretation $\mathcal{I}$, it takes an individual $d \in \Delta^{\mathcal{I}}$ and an $\mathcal{FL}_0$ concept C as input, and outputs a value in S expressing how far is d from belonging to $C^{\mathcal{I}}$. Membership distance functions are required to satisfy two properties, as stated in the following definition.

Definition 2. Given a linearly ordered set with minimum $(S, \leq, \mathfrak{m})$, a *membership distance function (mdf)* m is a family of functions containing for every interpretation $\mathcal{I}$ a function $m^{\mathcal{I}} : \Delta^{\mathcal{I}} \times \mathcal{C}_{\mathcal{FL}_0} \to S$, such that m satisfies the following (for all $\mathcal{FL}_0$ concepts C and D):

M1: for all interpretations $\mathcal{I}$ and all $d \in \Delta^{\mathcal{I}}$: $d \in C^{\mathcal{I}} \Leftrightarrow m^{\mathcal{I}}(d, C) = \mathfrak{m}$,

M2: $C \equiv D \Leftrightarrow m^{\mathcal{I}}(d, C) = m^{\mathcal{I}}(d, D)$ for all interpretations $\mathcal{I}$ and all $d \in \Delta^{\mathcal{I}}$.

The first property expresses the intuition that membership distance functions generalize the notion of classical membership. Regarding *M2*, it requires *equivalence invariance*, which means that m should behave the same for concepts that are equivalent, regardless of their syntactic definition. We are now ready to define the syntax and semantics of our family of logics $\mathcal{FL}_0at(m)$. Given the sets $\mathsf{N_C}$ and $\mathsf{N_R}$ of concept and role names, the set of $\mathcal{FL}_0at(m)$ concepts is defined as follows:

$$\widehat{C} ::= \top \mid A \mid \widehat{C} \sqcap \widehat{C} \mid \forall r.\widehat{C} \mid E_{\bowtie s},$$

where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, $\bowtie \in \{<, \leq, >, \geq\}$, $s \in S$, E is an $\mathcal{FL}_0$ concept description, and $\widehat{C}$ is a $\mathcal{FL}_0at(m)$ concept description. Concepts of the form $E_{\bowtie s}$ are called threshold concepts.

The semantics of $\mathcal{FL}_0at(m)$ concept descriptions is defined completely analogously to the semantics of classical $\mathcal{FL}_0$ concepts, but additionally using the parameter distance function m to interpret threshold concepts in the following way:

$$[E_{\bowtie s}]^{\mathcal{I}} := \{d \in \Delta^{\mathcal{I}} \mid m^{\mathcal{I}}(d, E) \bowtie s\}.$$

The following is a direct consequence of requiring property *M1*.

Proposition 3. *For every $\mathcal{FL}_0$ concept description C it holds that $C_{\leq \mathfrak{m}} \equiv C$ and $C_{>\mathfrak{m}} \equiv \neg C$, where the semantics of negation is $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$.*

Essentially, this tells us that the addition of threshold concepts allows for expressing the negation of $\mathcal{FL}_0$ concept descriptions. Therefore, differently from $\mathcal{FL}_0$, there are unsatisfiable $\mathcal{FL}_0at(m)$ concept descriptions, e.g., $A \sqcap A_{>\mathfrak{m}}$.

3.2 Membership distance functions and TBoxes

An important aspect when using DLs is to have the possibility to formulate (and take into account) terminological knowledge, which in DLs is expressed by GCIs in a TBox. For the family of logics $\mathcal{FL}_0\mathrm{at}(m)$, the most simple and natural form of such TBox is a plain $\mathcal{FL}_0$ TBox. For instance, we can define the $\mathcal{FL}_0at(m)$ concept descriptions $\forall r.(\forall s.A)_{<s}$ and $\forall r.(\forall r.B)_{<s}$ w.r.t. the following TBox:

$$\mathcal{T} := \{\forall s.A \sqsubseteq \forall r.B, \forall r.B \sqsubseteq \forall s.A\}.$$

Note that, although $\forall s.A \not\equiv \forall r.B$, they are actually equivalent modulo $\mathcal{T}$. Therefore, it should be the case that $(\forall s.A)_{<s} \equiv_{\mathcal{T}} (\forall r.B)_{<s}$ and $\forall r.(\forall s.A)_{<s} \equiv_{\mathcal{T}} \forall r.(\forall r.B)_{<s}$ hold in any particular logic $\mathcal{FL}_0at(m)$. However, this will not always be the case, since the semantics of $(\forall s.A)_{<s}$ and $(\forall r.B)_{<s}$ depends on the distance function m, but m need not take into account the GCIs in $\mathcal{T}$. Hence, to properly define the semantics of $\mathcal{FL}_0at(m)$ w.r.t. an $\mathcal{FL}_0$ TBox $\mathcal{T}$, we need to make m aware of the GCIs in $\mathcal{T}$. To this end, we slightly extend the definition of membership distance functions given in Definition 2 to a larger family of functions.

Definition 4. Given a linearly ordered set with minimum $(S, \leq, \mathfrak{m})$, a *membership distance function (mdf)* m is a family of functions containing for every $\mathcal{FL}_0$ TBox $\mathcal{T}$ and model $\mathcal{I}$ of $\mathcal{T}$ a function $m^{\mathcal{I},\mathcal{T}} : \Delta^{\mathcal{I}} \times \mathcal{C}_{\mathcal{FL}_0} \to S$, such that m satisfies the following (for all TBoxes $\mathcal{T}$ and $\mathcal{FL}_0$ concepts C and D):

$$M1' : \text{for all } \mathcal{I} \models \mathcal{T} \text{ and all } d \in \Delta^{\mathcal{I}} : \ d \in C^{\mathcal{I}} \Leftrightarrow m^{\mathcal{I},\mathcal{T}}(d, C) = \mathfrak{m},$$
$$M2' : C \equiv_{\mathcal{T}} D \Leftrightarrow m^{\mathcal{I},\mathcal{T}}(d, C) = m^{\mathcal{I},\mathcal{T}}(d, D) \text{ for all } \mathcal{I} \models \mathcal{T} \text{ all } d \in \Delta^{\mathcal{I}}.$$

Hence, in the presence of an $\mathcal{FL}_0$ TBox $\mathcal{T}$, the semantics of $\mathcal{FL}_0at(m)$ concepts is defined by using $m^{\mathcal{I},\mathcal{T}}$ to interpret threshold concepts $E_{\bowtie s}$ in any model $\mathcal{I}$ of $\mathcal{T}$. In the next sections, we will provide more concrete ways of defining distance functions w.r.t. $\mathcal{FL}_0$ TBoxes.

Finally, we would like to be able to state GCIs containing threshold concepts, e.g., $\forall r.(\forall s.A)_{\geq s} \sqsubseteq \forall r.B$. However, as pointed out in [BF16] for the family of threshold logics $\tau\mathcal{EL}(m)$, obtaining an appropriate semantics for TBoxes containing threshold concepts is not entirely trivial. We will consider this in future work.

4 Membership distance functions for $\mathcal{FL}_0$

In this section, we introduce a general framework to define membership distance functions for $\mathcal{FL}_0$. To this end, we exploit the connection between $\mathcal{FL}_0$ and formal languages to obtain a way of defining distance functions in terms of *language containment distances*, which are functions that compare tuples of languages. Finally, we describe a particular form that such containment distances can have, and provide concrete examples that can be derived from it.

4.1 Using tuples of languages to define membership distance functions

The idea of using tuples of languages to approximate the semantics of $\mathcal{FL}_0$ is not new. In [RS15, BMO16, BFM17], the authors exploit the fact that $\mathcal{FL}_0$ concepts can be represented as tuples of languages, to use these tuples to define *concept comparison measures (CCMs)* between $\mathcal{FL}_0$ concepts. Intuitively, a CCM generalizes classical equivalence (subsumption), by assigning to a pair of concepts (C, D) a degree to which equivalence (subsumption) between C and D is satisfied. Such measures are defined in [RS15, BMO16, BFM17] by using the following general approach:[1]

1. Translate the concepts C and D into the tuples of languages $\mathcal{L}_\mathcal{T}(C)$ and $\mathcal{L}_\mathcal{T}(D)$.
2. Compare the tuples $\mathcal{L}_\mathcal{T}(C)$ and $\mathcal{L}_\mathcal{T}(D)$ by using a function $\mathfrak{c}$ that assigns values from a numerical domain to tuples of languages.
3. The value $\mathfrak{c}(\mathcal{L}_\mathcal{T}(C), \mathcal{L}_\mathcal{T}(D))$ is then used to define the value of the CCM on C and D.

Since crisp membership in $\mathcal{FL}_0$ can also be characterized by using tuples of languages (see Theorem 1), we employ a similar approach to define membership distance functions for $\mathcal{FL}_0$. More precisely, we define membership distance functions as functions that compare tuples of languages, i.e., $m^{\mathcal{I},\mathcal{T}}(d, C)$ is defined by comparing the tuples $\mathcal{L}_\mathcal{T}(C)$ and $\mathcal{L}_\mathcal{I}(d)$. Clearly, not every function comparing tuples of languages yields a membership distance function, i.e., a family of functions satisfying the properties *M1′* and *M2′*. For this reason, we introduce the notion of *language containment distance (lcd)*, which is formally defined as follows.

Definition 5. Let $\mathfrak{l} = (S, \leq, \mathfrak{m})$ be a linearly ordered set with minimum $\mathfrak{m}$. In addition, let Σ be an alphabet and ℓ a positive integer. An *ℓ-language containment distance over Σ and $\mathfrak{l}$ (ℓ-lcd)* is a function $\mathfrak{c} : (2^{\Sigma^*})^\ell \times (2^{\Sigma^*})^\ell \to S$ that satisfies the property

$$\mathfrak{c}((L_1, \ldots, L_\ell), (M_1, \ldots, M_\ell)) = \mathfrak{m} \iff L_i \subseteq M_i \text{ for all } i, 1 \leq i \leq \ell. \tag{1}$$

We can now define a mechanism that, given $\mathsf{N_C} = \{A_1, \ldots, A_n\}$ and an n-lcd $\mathfrak{c}$ over $\Sigma = \mathsf{N_R}$ and $\mathfrak{l} = (S, \leq, \mathfrak{m})$, yields a distance function $m_\mathfrak{c}$ for $\mathcal{FL}_0$ concepts defined over $\mathsf{N_C}$ and $\mathsf{N_R}$.

Definition 6. Let $\mathfrak{c}$ be a n-lcd over $\mathsf{N_R}$ and $\mathfrak{l} = (S, \leq, \mathfrak{m})$. Then, for each $\mathcal{FL}_0$ TBox $\mathcal{T}$ and interpretation $\mathcal{I}$ that is a model of $\mathcal{T}$, the function $m_\mathfrak{c}^{\mathcal{I},\mathcal{T}} : \Delta^\mathcal{I} \times \mathcal{C}_{\mathcal{FL}_0} \mapsto S$ is defined as:

$$m_\mathfrak{c}^{\mathcal{I},\mathcal{T}}(d, C) = \mathfrak{c}\big(\mathcal{L}_\mathcal{T}(C), \mathcal{L}_\mathcal{I}(d)\big).$$

[1] In [RS15, BMO16], CCMs are only defined w.r.t. the empty TBox, i.e., by using $\mathcal{L}_\emptyset(C)$ and $\mathcal{L}_\emptyset(D)$.

The following lemma shows that the family of functions $m_{\mathfrak{c}} = \{m_{\mathfrak{c}}^{\mathcal{I},\mathcal{T}} \mid \mathcal{T} \text{ is an } \mathcal{FL}_0 \text{ TBox}, \mathcal{I} \models \mathcal{T}\}$ induced by an lcd $\mathfrak{c}$ satisfies the required properties, i.e., *M1′* and *M2′*, and hence the above framework can be used to define membership distance functions.

Lemma 7. *Let $\mathfrak{c}$ be an n-lcd over $\mathsf{N_R}$ and $\mathfrak{l} = (S, \leq, \mathfrak{m})$. Then, $m_{\mathfrak{c}}$ is a membership distance function.*

Proof. It suffices to prove that $m_{\mathfrak{c}}$ satisfies properties *M1′* and *M2′* from Definition 4.

- For *M1′*, let $\mathcal{T}$ be an $\mathcal{FL}_0$ TBox, $\mathcal{I}$ a model of $\mathcal{T}$, $d \in \Delta^{\mathcal{I}}$, $C \in \mathcal{C}_{\mathcal{FL}_0}$. Then
  $d \in C^{\mathcal{I}} \iff \mathcal{L}_{\mathcal{T}}(C, A_i) \subseteq \mathcal{L}_{\mathcal{I}}(d, A_i)$ for every $A_i \in \mathsf{N_C}$ by Theorem 1 $\iff m_{\mathfrak{c}}^{\mathcal{I},\mathcal{T}}(d, C) = \mathfrak{c}((\mathcal{L}_{\mathcal{T}}(C, A_1), \ldots, \mathcal{L}_{\mathcal{T}}(C, A_n)), (\mathcal{L}_{\mathcal{I}}(d, A_1), \ldots, \mathcal{L}_{\mathcal{I}}(d, A_n))) = \mathfrak{m}$ by the definition of lcd.
- For *M2′*, first assume that for a given $\mathcal{FL}_0$ TBox $\mathcal{T}$ and $C, D \in \mathcal{C}_{\mathcal{FL}_0}$ we have that $C \equiv_{\mathcal{T}} D$. Then $\mathcal{L}_{\mathcal{T}}(C) = \mathcal{L}_{\mathcal{T}}(D)$ and thus for any model $\mathcal{I}$ of $\mathcal{T}$ and $d \in \Delta^{\mathcal{I}}$ we have that $m_{\mathfrak{c}}^{\mathcal{I},\mathcal{T}}(d, C) = \mathfrak{c}(\mathcal{L}_{\mathcal{T}}(C), \mathcal{L}_{\mathcal{I}}(d)) = \mathfrak{c}(\mathcal{L}_{\mathcal{T}}(D), \mathcal{L}_{\mathcal{I}}(d)) = m_{\mathfrak{c}}^{\mathcal{I},\mathcal{T}}(d, D)$.
  For the opposite direction, assume that $C \not\equiv_{\mathcal{T}} D$. This means that there exists a model $\mathcal{I}_0$ s.t. $C^{\mathcal{I}_0} \neq D^{\mathcal{I}_0}$, which w.l.o.g. means that there exists $d \in \Delta^{\mathcal{I}_0}$ s.t. $d \in C^{\mathcal{I}_0}$ and $d \notin D^{\mathcal{I}_0}$. As a result, by *M1* we have that $m_{\mathfrak{c}}^{\mathcal{I}_0,\mathcal{T}}(d, C) = \mathfrak{m}$ but $m_{\mathfrak{c}}^{\mathcal{I}_0,\mathcal{T}}(d, D) \neq \mathfrak{m}$, hence the proof is complete.

□

4.2 Examples of language containment distances

One way to define ℓ-language containment distances is, given tuples $(L_1, \ldots, L_\ell)$ and $(M_1, \ldots, M_\ell)$, to use a 1-language containment distance to compare each pair of languages (L_i, M_i), and then apply an appropriate function that combines the obtained ℓ values into a single one [BMO16]. A function $f : S^\ell \to S$ is called an ℓ-ary *combining function* if it is *commutative:* $f(a_1, \ldots, a_\ell) = f(a_{\pi(1)}, \ldots, a_{\pi(\ell)})$ for all permutations π of indices $1, \ldots, \ell$, *monotone:* $a_1 \leq b_1, \ldots, a_\ell \leq b_\ell \implies f(a_1, \ldots, a_\ell) \leq f(b_1, \ldots, b_\ell)$, and $\mathfrak{m}$*-closed:* $f(a_1, \ldots, a_\ell) = \mathfrak{m} \iff a_1 = \cdots = a_\ell = \mathfrak{m}$.

For instance, for the interval $[0, 1]$ and the set of non-negative reals, with $\mathfrak{m} = 0$ and the usual order, examples of combining functions are the *maximum*, *average*, and the *sum* function.[2]

The following is an easy consequence of the properties required for combining functions and 1-language containment distances.

Lemma 8. *Let $\mathfrak{c}^1$ be a 1-language containment distance over Σ and $\mathfrak{l}$, and f an ℓ-ary combining function over S. Further, let $\mathfrak{c} : (2^{\Sigma^*})^\ell \times (2^{\Sigma^*})^\ell \to S$ be defined as:*

$$\mathfrak{c}((L_1, \ldots, L_\ell), (M_1, \ldots, M_\ell)) := f(\mathfrak{c}^1(L_1, M_1), \ldots, \mathfrak{c}^1(L_\ell, M_\ell)).$$

Then, $\mathfrak{c}$ is an ℓ-language containment distance over Σ and $\mathfrak{l}$.

We now provide some examples of ℓ-lcds that are obtained by using 1-language containment distances and ℓ-ary combining functions. These are defined over the set of non-negative reals.

- $\mathfrak{c}_0(L, M) = 0$ if $L \subseteq M$ and 1 otherwise. This is the simplest form of a 1-lcd, since it simply checks whether the first language is contained in the second one or not. It can be extended to an ℓ-lcd by using maximum, sum, or average:

[2] For the $[0, 1]$ interval, *sum* should be modified to *bounded sum*, i.e., $bs(a_1, \ldots, a_\ell) := \min(\sum_{i=1}^{\ell} a_i, 1)$.

- $\bar{\mathfrak{c}_0}\big((L_1,\dots,L_\ell),(M_1,\dots,M_\ell)\big) = \max(\mathfrak{c}_0(L_1,M_1),\dots,\mathfrak{c}_0(L_\ell,M_\ell))$,
- $\widehat{\mathfrak{c}_0}\big((L_1,\dots,L_\ell),(M_1,\dots,M_\ell)\big) = \sum_{i=1}^{n} \mathfrak{c}_0(L_i,M_i)$,
- $\tilde{\mathfrak{c}_0}\big((L_1,\dots,L_\ell),(M_1,\dots,M_\ell)\big) = \frac{1}{n}\sum_{i=1}^{\ell} \mathfrak{c}_0(L_i,M_i)$.

Intuitively, the first function checks whether containment is violated in any of the language pairs, the second one counts in how many of them a violation occurs, while the last one gives the percentage of pairs in which a violation of containment occurs.

- $\mathfrak{c}_1(L,M) = 2^{-m}$, where $m = \min\{|w| \mid w \in L \setminus M\}$. Obviously, if $L \subseteq M$ then $L \setminus M = \emptyset$, and hence $m = +\infty$, meaning that $c_1(L,M) = 0$. This function searches for the shortest word w that occurs in L but not in M, and assigns a greater value to shorter ones than longer ones. Once again, we can extend $\mathfrak{c}_1$ to an ℓ-lcd by using any of the functions described for $\mathfrak{c}_0$. If we use maximum, we are essentially looking for the shortest violation overall, with summing we are aggregating the penalties for the violations in the different pairs, while taking the average does not hold much intuitive meaning.
- $\mathfrak{c}_2(L,M) = \mu(L \setminus M)$, where $\mu(K) = \sum_{w\in K}(2|\Sigma|)^{-|w|}$. This function takes all violations into account, but longer words are penalized less than shorter ones. Extending $\mathfrak{c}_2$ to an ℓ-lcd by applying maximum yields a containment distance that searches for the "largest" difference in any pair of languages, whereas summing over the different pairs of languages corresponds to taking into account all the differences that occur. Finally, taking the average has the obvious meaning.

It is easy to verify that, by construction, $\mathfrak{c}_0, \mathfrak{c}_1, \mathfrak{c}_2$ are indeed 1-lcds, since they only output the value $\mathfrak{m} = 0$ iff the language in their first argument is contained in the one in the second.

Finally, since the main reason we employ lcds is to define membership distance functions, let us examine the meaning of these functions when applied to $\mathcal{L}_\mathcal{T}(C)$ and $\mathcal{L}_\mathcal{I}(d)$. It should be clear that if w is a violation of containment, this means that $C \sqsubseteq_\mathcal{T} \forall w.A$ (for some $A \in \mathsf{N_C}$), while $d \notin (\forall w.A)^\mathcal{I}$. As a result, $\mathfrak{c}_0$ (extended with the max function) simply checks whether $d \in C^\mathcal{I}$ (see Theorem 1), outputting 0, or not, outputting 1. Furthermore, the idea behind $\mathfrak{c}_1$ and $\mathfrak{c}_2$ that longer violations should count less than shorter ones corresponds to the view that differences "closer" to d are more important than ones further away.

5 Computability

In the previous section, we described how to reduce the definition of membership distance functions to defining measures that compare tuples of languages: $m^{\mathcal{I},\mathcal{T}}(d,C)$ corresponds to a value that results from comparing the tuples $\mathcal{L}_\mathcal{T}(C)$ and $\mathcal{L}_\mathcal{I}(d)$. In this section, we first demonstrate how to compute and (finitely) represent these tuples of (potentially) infinite languages, and how to assign such a value by making use of tree automata with weights. To this end, after establishing a correspondence between tuples of languages and infinite trees, we exhibit how these tuples can be computed and encoded using looping tree automata (LTAs) accepting the corresponding trees. Subsequently, we discuss how weighted looping tree automata perform the function of assigning values to trees and how they can be combined with the aforementioned LTAs. Overall, we obtain a concrete way to define membership distance functions that can be computed.

5.1 From tuples of languages to trees

The idea of representing tuples of languages using infinite trees has appeared in [BN01, BO13, Pen15, BFM17]. Initially, we provide the formal definition of infinite trees, before we discuss

how they can be used to represent tuples of languages.

Definition 9. Let $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$be a non-empty, finite set of symbols. Given a finite set of labels L, an L-labeled Σ-tree is a mapping $t : \Sigma^* \to L$ that assigns a label $t(w) \in L$ to every node $w \in \Sigma^*$. The set of all L-labeled Σ-trees is denoted as $T^{\omega}_{\Sigma,L}$.

Intuitively, the nodes of a Σ-tree t correspond to finite words in Σ^*, where the empty word ε is represented by the root of t and every node w has k children corresponding to the words $w\sigma_1, \ldots, w\sigma_k$. Furthermore, if we label each node of the tree with either 0 or 1, we can define a language over Σ by considering the words $w \in \Sigma$ for which $t(w) = 1$. If the labels were pairs of 0s and 1s, then two languages can be defined, one for each component, and so on.

More generally, the following mapping makes the connection between tuples of languages and infinite trees explicit.

Definition 10. Let Σ be a finite set of symbols and $\ell \in \mathbb{N}$. We define the mapping $\gamma_\ell : \left(2^{\Sigma^*}\right)^\ell \to T^{\omega}_{\Sigma,\{0,1\}^\ell}$ as follows: given a tuple of languages $\mathbf{L} = (L_1, \ldots, L_\ell)$ over Σ, we set $\gamma_\ell(\mathbf{L}) := t_{\mathbf{L}}$ where $t_{\mathbf{L}} : \Sigma^* \to \{0,1\}^\ell$ is the Σ-tree such that

$$t_{\mathbf{L}}(w) := (x_1, \ldots, x_\ell), \text{ where } x_i = 1 \text{ iff } w \in L_i \quad (\text{for all } w \in \Sigma^*).$$

It is easy to see that γ_ℓ is a *bijection* between tuples of ℓ languages over the alphabet Σ and $\{0,1\}^\ell$-labeled Σ-trees. Given $t \in T^{\omega}_{\Sigma,\{0,1\}^\ell}$, the inverse function yields the tuple $\gamma_\ell^{-1}(t) = (L_1, \ldots, L_\ell)$ where L_i consists of the words w for which the i-th component of $t(w)$ is 1.

In particular, if we fix $\Sigma = \mathsf{N_R}$ and a linear order between the concept names in $\mathsf{N_C} = (A_1, \ldots, A_n)$, the tuple of languages $\mathcal{L}_\mathcal{T}(C)$ (likewise for $\mathcal{L}_\mathcal{I}(d)$) can be represented by the $\mathsf{N_R}$-tree $t_{\mathcal{L}_\mathcal{T}(C)}$ with labels from $\{0,1\}^n$ defined as:

$$t_{\mathcal{L}_\mathcal{T}(C)}(w) := (x_1, \ldots, x_n), \text{ where } x_i = 1 \text{ iff } w \in \mathcal{L}_\mathcal{T}(C, A_i) \quad (\text{for all } w \in \Sigma^*).$$

Note that $\mathcal{L}_\mathcal{T}(C)$ and $\mathcal{L}_\mathcal{I}(d)$ can be put together in a single tree, by using labels of length $2n$.

So far, we have seen how concept descriptions and individuals in an interpretation can be represented as tuples of languages, which in turn correspond to (infinite) trees. Next, we need to represent said trees in a finite way. Obviously, this is not always possible for infinite trees. However, for the needs of our work, the relevant trees are *regular* (see [BFM17, Pen15] for $\mathcal{L}_\mathcal{T}(C)$ and the proof of Prop. 13 for $\mathcal{L}_\mathcal{I}(d)$ when the interpretation $\mathcal{I}$ is finite). There are different ways to represent regular trees in a finite way [Tho90]. Here, we use *looping tree automata* for this purpose.

Definition 11 (Looping tree automaton (LTA))**.** A *looping tree automaton* is a tuple $\mathcal{A} = (\Sigma, Q, L, \Delta, I)$ where $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$ is a finite set of symbols, Q is a finite set of states, L is a finite set of labels, $\Delta \subseteq Q \times L \times Q^k$ is the transition relation and $I \subseteq Q$ is a set of initial states. A run of this automaton on a tree $t \in T^{\omega}_{\Sigma,L}$ is a Q-labeled Σ-tree $\rho : \Sigma^* \to Q$ such that: $\rho(\varepsilon) \in I$ and $(\rho(w), t(w), \rho(w\sigma_1), \ldots, \rho(w\sigma_k)) \in \Delta$ for all $w \in \Sigma^*$. The tree language $\mathcal{L}(\mathcal{A})$ recognized by $\mathcal{A}$ is the set of all trees $t \in T^{\omega}_{\Sigma,L}$ such that $\mathcal{A}$ accepts t, i.e., $\mathcal{A}$ has a run on t. If $\mathcal{L}(\mathcal{A}) = \{t_0\}$, then we say that $\mathcal{A}$ is *representing* the tree t_0.

Essentially, if an LTA is representing a single tree, it is a finite representation of this tree. In [BFM17] it was proved that regular trees can always be represented by an LTA. In particular, for the case of $\mathcal{L}_\mathcal{T}(C)$, the following result was shown in [Pen15].

Proposition 12 ([Pen15])**.** *Let C be an $\mathcal{FL}_0$ concept description and $\mathcal{T}$ an $\mathcal{FL}_0$ TBox. Then, one can construct an LTA that represents $t_{\mathcal{L}_\mathcal{T}(C)}$ in time exponential in the size of C and $\mathcal{T}$.*

For $\mathcal{L}_{\mathcal{I}}(d)$, however, one has to be more careful. For *infinite* interpretations, the languages $\mathcal{L}_{\mathcal{I}}(d, A)$ need not be regular (and hence also the tree $t_{\mathcal{L}_{\mathcal{I}}(d)}$). Still, for finite interpretations these languages are always regular (as Prop. 13 below shows), and hence can be represented by deterministic finite automata. Intuitively, we can view a finite interpretation as a finite automaton, where the individuals are states and role connections correspond to transitions. It is then not difficult to verify that for an individual d and a concept name A one can recursively follow the paths in the graph to check whether $d \in (\forall w.A)^{\mathcal{I}}$ or not. The language $\mathcal{L}_{\mathcal{I}}(d, A)$ will be the solution of a set of language equations. Overall, we have the following result.

Proposition 13. *Let $\mathsf{N_C} = \{A_1, \ldots, A_n\}$. Given a finite interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ and $d \in \Delta^{\mathcal{I}}$ one can construct DFAs that recognize the languages $\mathcal{L}_{\mathcal{I}}(d, A_1), \ldots, \mathcal{L}_{\mathcal{I}}(d, A_n)$ and a looping tree automaton that is representing the tree $t_{\mathcal{L}_{\mathcal{I}}(d)}$ in time exponential in the size of $\mathcal{I}$ and $\mathsf{N_C}$.*

Proof. Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a finite interpretation with $\Delta^{\mathcal{I}} = \{d_1, \ldots, d_n\}$. Denote $r(d) = \{e \in \Delta^{\mathcal{I}} \mid (d, e) \in r^{\mathcal{I}}\}$ for every $r \in N_R$ and $d \in \Delta^{\mathcal{I}}$ and $\ell(d) = \{A \in N_C \mid d \in A^{\mathcal{I}}\}$. The characteristic function $\delta_{x,X}$ is defined

$$\delta_{x,X} = \begin{cases} \{\varepsilon\} & \text{if } x \in X \\ \emptyset & \text{otherwise} \end{cases}$$

Recall that

$$\mathcal{L}_{\mathcal{I}}(d, A) = \{w \in N_R \mid d \in (\forall w.A)^{\mathcal{I}}\}.$$

It is easy to verify that

$$\mathcal{L}_{\mathcal{I}}(d, A) = \delta_{A,\ell(d)} \cup \bigcup_{r \in N_R} r \bigcap_{d' \in r(d)} \mathcal{L}_{\mathcal{I}}(d', A).$$

Note that, in case $r(d) = \emptyset$, the empty intersection corresponds to $\mathsf{N_R}^*$.

This observation allows us to compute the languages $\mathcal{L}_{\mathcal{I}}(d, A)$, given the following claim.

Claim. The tuple $\mathbf{L} = (\mathcal{L}_{\mathcal{I}}(d_1, A), \ldots, \mathcal{L}_{\mathcal{I}}(d_n, A))$ is the least solution of the system of language equations

$$X_i = \delta_{A,\ell(d_i)} \cup \bigcup_{r \in N_R} r \bigcap_{d_j \in r(d_i)} X_j \tag{2}$$

Proof of the Claim. By the previous observation, $\mathbf{L}$ is a solution of the system of equations (2). Assume that the tuple $(M_1, \ldots, M_n)$ is another solution. By induction on the length of the words, we will show that $w \in \mathcal{L}_{\mathcal{I}}(d_i, A) \implies w \in M_i$, and thus $\mathbf{L}$ is the least solution.

- If $w = \varepsilon$, we get that

$$\begin{aligned} \varepsilon \in \mathcal{L}_{\mathcal{I}}(d_i, A) &\implies d_i \in A^{\mathcal{I}} \implies A \in \ell(d_i) \\ &\implies \delta_{A,\ell(d_i)} = \{\varepsilon\} \implies \varepsilon \in M_i. \end{aligned}$$

- If $w = rv$ we get that for all r-successors d_j of d_i, $d_j \in (\forall v.A)^{\mathcal{I}}$, and thus $v \in \mathcal{L}_{\mathcal{I}}(d_j, A)$. By the induction hypothesis, $v \in M_j$ for all r-successors d_j of d_i. Hence, $rv \in M_i$, and the proof of the Claim is complete.

To effectively construct automata that recognize the languages of the least solution, one can use the results of [BO13] to build deterministic finite automata (DFAs) in ExpTime.

The DFAs for the languages $\mathcal{L}_{\mathcal{I}}(d, A)$ for $A \in \mathsf{N_C}$ can then be combined into a single LTA representing the tuple of languages, reversing the technique that is used in [Pen15] to extract DFAs for each language $\mathcal{L}_{\mathcal{T}}(C, A)$, $A \in \mathsf{N_C}$ from the LTA representing $t_{\mathcal{L}_{\mathcal{T}}(C)}$. □

5.2 Assigning values to trees

We now want to assign values from a proper (numerical) domain to trees (that correspond to tuples of languages). This is exactly the operation of infinitary tree series, for which the assigned values are usually elements of a *semiring*, i.e., a domain S equipped with two operations, the one traditionally called "addition" and the other "multiplication", that satisfy certain mathematical properties. Formally, an *infinitary tree series* h over the alphabet L and semiring $\mathcal{S}$ is a mapping $h : T^{\omega}_{\Sigma,L} \to S$. The class of all infinitary tree series over L and $\mathcal{S}$ is denoted by $\mathcal{S}\langle\langle T^{\omega}_{\Sigma,L}\rangle\rangle$.

One way to (finitely) define such series, is using a *weighted looping tree automaton* (wLTA) ([BFM17]). Intuitively, a wLTA $\mathcal{M}$ attributes a *weight*, i.e., a value from a semiring to every transition, and "multiplies" all the weights accumulated during a certain run on a tree. Finally, it "sums" the weights of all the runs to determine the value that will be assigned to the input tree. For this purpose, it is clear that the underlying semiring should admit suitable infinite sums and products. In *totally complete commutative semirings*, addition and multiplication can be suitably extended to infinite sums and countably infinite products (see [Rah07] for formal definitions).

Furthermore, since we want the output values to be used for membership distance functions, we require that the domain of $\mathcal{S}$ is also equipped with a linear order and has a minimum element $\mathfrak{m}$. This is not a heavy restriction, since most numeric semirings are already equipped with such an order. Examples are the semiring of natural numbers $(\mathbb{N} \cup \{+\infty\}, +, \cdot, 0, 1)$, the tropical semiring $Trop = (\mathbb{N} \cup \{+\infty\}, \min, +, +\infty, 0)$, and its real counterpart $\mathbb{R}_{inf} = (\mathbb{R}_{\geq 0} \cup \{+\infty\}, \inf, +, +\infty, 0)$, all equipped with the usual order and 0 as the minimum element.

The infinitary tree series $||\mathcal{M}|| \in \mathcal{S}\langle\langle T^{\omega}_{\Sigma,L}\rangle\rangle$ defined by the wLTA $\mathcal{M}$ is called the *behavior* of the wLTA $\mathcal{M}$. It assigns to every tree $t \in T^{\omega}_{\Sigma,L}$ a value $(||\mathcal{M}||, t)$. As demonstrated in [BFM17], wLTAs can be used to define functions over tuples of languages, viewed as infinite trees. In fact, the language containment distances described in Section 4 are variations of the functions defined in [BFM17], and they can also be defined by a wLTA by using similar constructions.

It is also shown in [BFM17] that, for certain semirings, a wLTA $\mathcal{M}$ can be combined with an LTA $\mathcal{A}$ representing a tree t in order to compute the value $(||\mathcal{M}||, t)$ in time polynomial in the size of $\mathcal{M}$ and $\mathcal{A}$. As a result, given a wLTA $\mathcal{M}$ defining a language containment distance $\mathfrak{c}$, the LTAs obtained from $\mathcal{L}_{\mathcal{T}}(C)$ and $\mathcal{L}_{\mathcal{I}}(d)$ can be combined with $\mathcal{M}$ in order to compute $m^{\mathcal{I},\mathcal{T}}_{\mathfrak{c}}(d, C)$, i.e., the value $(||\mathcal{M}||, t_{C,\mathcal{I}})$ where $t_{C,\mathcal{I}}$ is the single tree representing $\mathcal{L}_{\mathcal{T}}(C)$ and $\mathcal{L}_{\mathcal{I}}(C)$. This "computable" family of wLTAs includes the wLTAs defining the language containment distances described in Section 4. Overall we obtain the following result.

Theorem 14. *Given a wLTA $\mathcal{M}$ that computes a language containment distance $\mathfrak{c}$, an $\mathcal{FL}_0$ TBox $\mathcal{T}$, a finite model $\mathcal{I}$ of $\mathcal{T}$, $d \in \Delta^{\mathcal{I}}$, and $C \in \mathcal{C}_{\mathcal{FL}_0}$, the value $m^{\mathcal{I},\mathcal{T}}_{\mathfrak{c}}(d, C)$ is computable. In particular, this holds for all concrete containment distances $\mathfrak{c}$ defined in Section 4.*

6 Conclusion and Future Work

We have introduced a family of DLs $\mathcal{FL}_0 at(m)$ that extends the DL $\mathcal{FL}_0$ with threshold concepts, whose semantics is defined by using a membership distance function m. We have demonstrated how membership of an indvidual in an $\mathcal{FL}_0$ concept can be characterized using formal languages, similarly to existing characterizations of subsumption and equivalence between $\mathcal{FL}_0$ concepts. Utilizing this characterization, we derived a framework for obtaining membership distance functions by employing functions that compare tuples of formal languages, namely language containment distances. Finally, we exhibited how weighted looping tree automata can be exploited to derive concrete and computable functions through our framework.

The natural continuation in this line of work is to study reasoning problems in $\mathcal{FL}_0 at(m)$ for particular membership distance functions. A powerful tool when reasoning in $\mathcal{FL}_0$ is the notion of a *functional model* of a concept description C [Pen15], i.e., an interpretation that has the shape of an infinite tree, where every node has exactly one r successor for every $r \in \mathsf{N_R}$, and the root of which belongs to (the interpretation of) C. Among others, subsumption ([Pen15]) and instance checking ([BMP18]) can be decided using said models, since every $\mathcal{FL}_0$ concept has a functional model. One could reasonably assume that a similar technique could be employed for $\mathcal{FL}_0 at(m)$. For example, in order to investigate satisfiability of a concept description, the search space could potentially be reduced to the set of functional models. However, this is not possible in $\mathcal{FL}_0 at(m)$, as a result of Proposition 3. More precisely, the concept description $\widehat{C} := \forall r.(A \sqcap A_{>\mathfrak{m}})$ is satisfiable but it requires that any individual d in the extension of $\widehat{C}$ has no r successors. As a result, this concept description has no functional model for any membership distance function. It would be interesting to investigate if a such an approach can be used in order to reason in this threshold setting.

References

[Baa96] Franz Baader. Using automata theory for characterizing the semantics of terminological cycles. *Ann. Math. Artif. Intell.*, 18(2-4):175–219, 1996.

[BBF15] Franz Baader, Gerhard Brewka, and Oliver Fernández Gil. Adding threshold concepts to the description logic $\mathcal{EL}$. In *Proc. of the 10th Int. Symp. on Frontiers of Combining Systems (FroCoS 2015)*, volume 9322 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 2015.

[BCM+03] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[BE16] Franz Baader and Andreas Ecke. Reasoning with prototypes in the description logic $\mathcal{ALC}$ using weighted tree automata. In *Language and Automata Theory and Applications - 10th International Conference, LATA 2016, Prague, Czech Republic, March 14-18, 2016, Proceedings*, volume 9618 of *Lecture Notes in Computer Science*, pages 63–75. Springer, 2016.

[BF16] Franz Baader and Oliver Fernández Gil. Extending the description logic $\tau\mathcal{EL}(deg)$ with acyclic TBoxes. In *Proc. of the 22nd Eur. Conf. on Artificial Intelligence (ECAI 2016)*, volume 285 of *Frontiers in Artificial Intelligence and Applications*, pages 1096–1104. IOS Press, 2016.

[BFM17] Franz Baader, Oliver Fernández Gil, and Pavlos Marantidis. Approximation in description logics: How weighted tree automata can help to define the required concept comparison measures in $\mathcal{FL}_0$. In *Proceedings of the 11th International Conference on Language and Automata Theory and Applications (LATA 2017)*, volume 10168 of *Lecture Notes in Computer Science*, pages 3–26. Springer, 2017.

[BHLS17] Franz Baader, Ian Horrocks, Carsten Lutz, and Uli Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.

[BMO16] Franz Baader, Pavlos Marantidis, and Alexander Okhotin. Approximate unification in the description logic $\mathcal{FL}_0$. In *Proc. of the 15th Eur. Conf. on Logics in Artificial Intelligence (JELIA'2016)*, volume 10021 of *Lecture Notes in Computer Science*, pages 49–63. Springer, 2016.

[BMP18] Franz Baader, Pavlos Marantidis, and Maximilian Pensel. The data complexity of answering instance queries in $\mathcal{FL}_0$. In Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis, editors, *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1603–1607. ACM, 2018.

[BN01] Franz Baader and Paliath Narendran. Unification of concept terms in description logics. *J. of Symbolic Computation*, 31(3):277–305, 2001.

[BO13] Franz Baader and Alexander Okhotin. On language equations with one-sided concatenation. *Fundamenta Informaticae*, 126(1):1–35, 2013.

[dFEL13] Claudia d'Amato, Nicola Fanizzi, Floriana Esposito, and Thomas Lukasiewicz. Representing uncertain concepts in rough description logics via contextual indiscernibility relations. In *Uncertainty Reasoning for the Semantic Web II, International Workshops URSW 2008-2010 Held at ISWC and UniDL 2010 Held at FLoC, Revised Selected Papers*, volume 7123 of *Lecture Notes in Computer Science*, pages 300–314. Springer, 2013.

[GKP+19] Pietro Galliani, Oliver Kutz, Daniele Porello, Guendalina Righetti, and Nicolas Troquard. On knowledge dependence in weighted description logic. In *GCAI 2019. Proceedings of the 5th Global Conference on Artificial Intelligence, Bozen/Bolzano, Italy, 17-19 September 2019*, volume 65 of *EPiC Series in Computing*, pages 68–80. EasyChair, 2019.

[Háj05] Petr Hájek. Making fuzzy description logic more general. *Fuzzy Sets and Systems*, 154(1):1–15, 2005.

[LWZ03] Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. A tableau algorithm for reasoning about concepts and similarity. In *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX 2003, Rome, Italy, September 9-12, 2003. Proceedings*, volume 2796 of *Lecture Notes in Computer Science*, pages 134–149. Springer, 2003.

[Pen15] Maximilian Pensel. An automata based approach for subsumption w.r.t. general concept inclusions in the description logic $\mathcal{FL}_0$. Master's thesis, Chair for Automata Theory, TU Dresden, Germany. See http://lat.inf.tu-dresden.de/research/mas., 2015.

[PKR+19] Daniele Porello, Oliver Kutz, Guendalina Righetti, Nicolas Troquard, Pietro Galliani, and Claudio Masolo. A toothful of concepts: Towards a theory of weighted concept combination. In *Proceedings of the 32nd International Workshop on Description Logics, Oslo, Norway, June 18-21, 2019*, volume 2373 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[PZ13] Rafael Peñaloza and Tingting Zou. Roughening the envelope. In *Frontiers of Combining Systems - 9th International Symposium, FroCoS 2013, Nancy, France, September 18-20, 2013. Proceedings*, volume 8152 of *Lecture Notes in Computer Science*, pages 71–86. Springer, 2013.

[Rah07] George Rahonis. Weighted muller tree automata and weighted logics. *J. Autom. Lang. Comb.*, 12(4):455–483, 2007.

[RS15] T. Racharak and B. Suntisrivaraporn. Similarity measures for $\mathcal{FL}_0$ concept descriptions from an automata-theoretic point of view. In *6th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES)*, pages 1–6, 2015.

[SKP07] Stefan Schlobach, Michel C. A. Klein, and Linda Peelen. Description logics with approximate definitions - precise modeling of vague concepts. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 557–562, 2007.

[Str01] Umberto Straccia. Reasoning within fuzzy description logics. *J. of Artificial Intelligence Research*, 14:137–166, 2001.

[STWZ07] Mikhail Sheremet, Dmitry Tishkovsky, Frank Wolter, and Michael Zakharyaschev. A logic for concepts and similarity. *J. of Logic and Computation*, 17(3):415–452, 2007.

[Tho90] Wolfgang Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 133–192. The MIT Press, 1990.

[Yen91] John Yen. Generalizing term subsumption languages to fuzzy logic. In John Mylopoulos and Raymond Reiter, editors, *Proceedings of the 12th International Joint Conference on Artificial Intelligence, 1991*, pages 472–477. Morgan Kaufmann, 1991.