The Concept Difference for $\mathcal{EL}$-Terminologies using Hypergraphs*

Andreas Ecke
Theoretical Computer Science
TU Dresden, Germany
ecke@tcs.inf.tu-dresden.de

Michel Ludwig
Theoretical Computer Science
TU Dresden, Germany
Center for Advancing Electronics Dresden
michel@tcs.inf.tu-dresden.de

Dirk Walther
Theoretical Computer Science
TU Dresden, Germany
Center for Advancing Electronics Dresden
dirk@tcs.inf.tu-dresden.de

ABSTRACT

Ontologies are used to represent and share knowledge. Numerous ontologies have been developed so far, especially in knowledge intensive areas such as the biomedical domain. As the size of ontologies increases, their continued development and maintenance is becoming more challenging as well. Detecting and representing semantic differences between versions of ontologies is an important task for which automated tool support is needed. In this paper we investigate the logical difference problem using a hypergraph representation of $\mathcal{EL}$-terminologies. We focus solely on the concept difference wrt. a signature. For computing this difference it suffices to check the existence of simulations between hypergraphs whereas previous approaches required a combination of different methods.

1. INTRODUCTION

Ontologies are widely used to represent domain knowledge. They contain specifications of objects, concepts and relationships that are often formalised using a logic-based language over a vocabulary that is particular to an application domain. Ontology languages based on description logics [2] have been widely adopted, e.g., description logics are underlying the Web Ontology Language (OWL) and its profiles.[1] Numerous ontologies have already been developed, in particular, in knowledge intensive areas such as the biomedical domain.[2] Ontologies constantly evolve, they are regularly extended, corrected and refined. As the size of ontologies increases, their continued development and maintenance becomes more challenging as well. For instance, the ontology SNOMED CT contains now definitions for about 400 000 terms, and the 'NCBI organismal classification' ontology even for about 850 000 terms. In particular, the need to have automated tool support for detecting and representing differences between versions of an ontology is growing in importance for ontology engineering. Current support from ontology editors, such as Protegé, SWOOP, OBO-Edit, and OntoView, is mostly based on syntactic differences and does not capture the semantic differences between ontologies. An early detection of possibly unwanted semantic changes can contribute to an error-resilient authoring process of ontologies.

The aim of this paper is to propose and investigate the logical difference problem using a hypergraph representation of ontologies. The logical difference problem was introduced in [7], where the logical difference is taken to be the set of queries formulated in a vocabulary of interest, called signature, that produce different answers when evaluated over ontologies that are to be compared. In this paper we concentrate on ontologies expressed as terminologies in the lightweight description logic $\mathcal{EL}$ [1, 3] and on queries that are concept inclusions formulated in $\mathcal{EL}$. Even though $\mathcal{EL}$-terminologies merely serve as a starting point for this investigation, we can illustrate the elegance of the hypergraph-based approach and the advantages over existing approaches to computing the logical difference. The relevance of $\mathcal{EL}$ is emphasised by the fact that many ontologies are largely formulated in $\mathcal{EL}$, notable examples being SNOMED CT and NCI.

An $\mathcal{EL}$-terminology can easily be translated into a directed hypergraph by taking the signature symbols as nodes and treating the axioms as hyperedges. For instance, the axiom $A \sqsubseteq \exists r.B$ is translated into the hyperedge $(\{x_A\}, \{x_r, x_B\})$, and the axiom $A \equiv B_1 \sqcap B_2$ into the three hyperedges $(\{x_A\}, \{x_{B_1}\})$, $(\{x_A\}, \{x_{B_2}\})$ and $(\{x_{B_1}, x_{B_2}\}, \{x_A\})$, where each node x_Y corresponds to the signature symbol Y, respectively. A feature of the translation of axioms into hyperedges is that all information about the axiom and the logical operators in it is preserved. We can actually treat the ontology and its hypergraph interchangeably. The existence of certain simulations between hypergraphs characterises the fact that the corresponding terminologies are logically equivalent and, thus, no logical difference exists. If no simulation ex-

*We thank the reviewers of the workshop DChanges 2013 for their comments. The authors acknowledge the support of the German Research Foundation (DFG), Andreas Ecke within GRK 1763 (QuantLA), and Michel Ludwig and Dirk Walther within the Resilience and Bio Path of the Cluster of Excellence 'Center for Advancing Electronics Dresden'.

[1] http://www.w3.org/TR/owl2-overview/

[2] http://bioportal.bioontology.org

DChanges 2013, September 10th, 2013, Florence, Italy.
ceur-ws.org Volume 1008, http://ceur-ws.org/Vol-1008/paper3.pdf

ists, we can directly extract the axioms responsible for the concept inclusion that witnesses the logical difference from the hypergraph.

The main advantages of the hypergraph-based approach to logical difference are: (i) an elegant algorithm for detecting the existence of concept differences (solely involving checking for simulations in hypergraphs), even for large or *cyclic* terminologies; (ii) a straightforward way to construct concept inclusions that witness the logical difference between two terminologies, even for *cyclic* terminologies; and (iii) a simple computation of explanations, i.e., sets of axioms that entail such concept inclusions. Currently, the algorithms implemented for detecting the logical difference work for large but acyclic terminologies such as SNOMED CT [5–7]. The algorithm in [6] can also handle "small" cyclic terminologies, but the concept inclusions witnessing a difference cannot easily be constructed using that algorithm.

The paper is organised as follows. We start by reviewing some notions regarding the description logic $\mathcal{EL}$, the logical difference problem, and ontology hypergraphs. In Section 3, we introduce two simulation notions, a forward and a backward simulation, one for each type of concept inclusion that may witness the logical difference between two terminologies. In each case we show that the existence of a simulation between two terminologies corresponds to the absence of difference witnesses. We analyse the computational complexity of checking for simulations, and we sketch how to construct counter-examples. In Section 4, we discuss previous approaches to computing the logical difference in [5] and explain the advantages of the hypergraph-based approach introduced in this paper. Finally we conclude the paper.

2. PRELIMINARIES

We start by briefly reviewing the lightweight description logic $\mathcal{EL}$ and some notions related to the logical difference, together with some basic results.

2.1 The Logic $\mathcal{EL}$

Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be mutually disjoint sets of concept names and role names. We assume these sets to be countably infinite. We typically use A, B to denote concept names and r to denote role names. The set of $\mathcal{EL}$*-concepts* C is defined inductively as:

- $\top$ and all concept names in $\mathsf{N_C}$ are $\mathcal{EL}$-concepts,
- if C, D are $\mathcal{EL}$-concepts, then $C \sqcap D$ and $\exists r.C$ are $\mathcal{EL}$-concepts, where $r \in \mathsf{N_R}$.

An $\mathcal{EL}$*-TBox* $\mathcal{T}$ is a finite set of axioms, where an axiom can be a *concept inclusion* $C \sqsubseteq D$, or a *concept equation* $C \equiv D$, where C, D range over $\mathcal{EL}$-concepts.

The semantics of $\mathcal{EL}$ is defined using interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where the domain $\Delta^{\mathcal{I}}$ is a non-empty set, and $\cdot^{\mathcal{I}}$ is a function mapping each concept name A to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ and every role name r to a binary relation $r^{\mathcal{I}}$ over $\Delta^{\mathcal{I}}$. The extension $C^{\mathcal{I}}$ of a concept C is defined inductively as follows: $\top^{\mathcal{I}} := \Delta^{\mathcal{I}}$, $(C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}$ and $(\exists r.C)^{\mathcal{I}} := \{x \in \Delta^{\mathcal{I}} \mid \exists y \in C^{\mathcal{I}} : (x, y) \in r^{\mathcal{I}}\}$. An interpretation $\mathcal{I}$ *satisfies* a concept C, an axiom $C \sqsubseteq D$ or $C \equiv D$ if, respectively, $C^{\mathcal{I}} \neq \emptyset$, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, or $C^{\mathcal{I}} = D^{\mathcal{I}}$. We write $\mathcal{I} \models \alpha$ if $\mathcal{I}$ satisfies the axiom α. An interpretation $\mathcal{I}$ *satisfies* a TBox $\mathcal{T}$ if $\mathcal{I}$ satisfies all axioms in $\mathcal{T}$; in this case, we say that $\mathcal{I}$ is a *model* of $\mathcal{T}$. An axiom α *follows* from a TBox $\mathcal{T}$, written $\mathcal{T} \models \alpha$, if for all models $\mathcal{I}$ of $\mathcal{T}$, we have that $\mathcal{I} \models \alpha$. Checking that $\mathcal{T} \models \alpha$ can be done in polynomial time in the size of $\mathcal{T}$ and α [1, 3].

A signature Σ is a finite set of symbols from $\mathsf{N_C}$ and $\mathsf{N_R}$. The signature $\mathsf{sig}(C)$, $\mathsf{sig}(\alpha)$ or $\mathsf{sig}(\mathcal{T})$ of the concept C, axiom α or TBox $\mathcal{T}$ is the set of concept and role names occurring in C, α or $\mathcal{T}$, respectively. An $\mathcal{EL}_\Sigma$-concept C is an $\mathcal{EL}$-concept such that $\mathsf{sig}(C) \subseteq \Sigma$.

Two TBoxes $\mathcal{T}$ and $\mathcal{T}'$ are *logically equivalent wrt. a signature* Σ, written $\mathcal{T} \equiv_\Sigma \mathcal{T}'$, if for all $\mathcal{EL}$-axioms α with $\mathsf{sig}(\alpha) \subseteq \Sigma$: $\mathcal{T} \models \alpha$ iff $\mathcal{T}' \models \alpha$. In other words, two TBoxes are logically equivalent wrt. a signature if the same axioms formulated in the signature follow from them. In this case, the TBoxes are also said to be Σ-*inseparable*. Conservative extensions are a special case of logical equivalence: for $\mathcal{T} \subseteq \mathcal{T}'$ and $\Sigma = \mathsf{sig}(\mathcal{T})$, $\mathcal{T}'$ is a conservative extension of $\mathcal{T}$ wrt. Σ iff $\mathcal{T} \equiv_\Sigma \mathcal{T}'$. Deciding the logical equivalence of $\mathcal{EL}$-TBoxes wrt. a signature is ExpTime-complete [9].

To be able to better deal with complex concepts in a TBox, we assume that there are no nested existential restrictions. We say that a TBox $\mathcal{T}$ is *flattened* if all conjunctions $C \sqcap D$ and existential restrictions $\exists r.E$ in $\mathcal{T}$ are such that C, D are concept names or conjunctions, and E is a concept name. We ignore the nesting of binary conjunctions and treat them as n-ary conjunctions of n concept names, where $n \geq 2$. The axioms of a flattened TBox are of the form $X \bowtie Y$, where $X, Y \in \{\top\} \cup \{B_1 \sqcap \cdots \sqcap B_n \mid n > 0, B_i \in \mathsf{N_C}\} \cup \{\exists r.A \mid r \in \mathsf{N_R}, A \in \mathsf{N_C}\}$ and $\bowtie \in \{\sqsubseteq, \equiv\}$. Any $\mathcal{EL}$-TBox can be flattened by appropriately replacing nested complex concepts C by fresh concept names X_C and adding concept equations $X_C \equiv C$ to the TBox that define the new symbols. It can be readily seen that this transformation is tractable and that it does not change the meaning of the original TBox. The following lemma makes this precise.

Lemma 1. For every $\mathcal{EL}$-TBox $\mathcal{T}$, there is a flattened $\mathcal{EL}$-TBox $\mathcal{T}$' of polynomial size in the size of $\mathcal{T}$ such that $\mathcal{T} \equiv_\Sigma \mathcal{T}'$ with $\Sigma = \mathsf{sig}(\mathcal{T})$.

For the remainder of the paper we assume that TBoxes are flattened.

2.2 Terminologies in Normal Form

An important motivating feature of $\mathcal{EL}$ is that it exhibits a low complexity for standard reasoning tasks. However, as we have seen above, deciding the logical equivalence of $\mathcal{EL}$-TBoxes wrt. a signature already requires exponential time.[3] To gain tractability for deciding the logical equivalence, TBoxes are restricted to a particular form as in [5, 7].

Definition 1. An $\mathcal{EL}$-TBox $\mathcal{T}$ is called an $\mathcal{EL}$*-terminology* if it satisfies the following conditions:

[3]Note that it is tractable to check the logical equivalence of two $\mathcal{EL}$-TBoxes without restricting the signature [1, 3].

- all concept inclusions and equations in $\mathcal{T}$ are of the form $A \sqsubseteq C$, $A \equiv C$, where A is a concept name, and
- no concept name A occurs more than once on the left-hand side of an axiom in $\mathcal{T}$.

The restriction to $\mathcal{EL}$-terminologies yields that deciding logical equivalence wrt. a signature becomes tractable [5, 7].

Definitions in terminologies can be cyclic, which may cause difficulties for reasoning algorithms. A terminology is cyclic if a concept name refers to itself along concept inclusions and equations. To be precise, for a terminology $\mathcal{T}$, let $\prec_\mathcal{T}$ be a binary relation over $\mathsf{N_C}$ such that $A \prec_\mathcal{T} B$ if there is an axiom of the form $A \sqsubseteq C$ or $A \equiv C$ in $\mathcal{T}$ such that $B \in \mathsf{sig}(C)$. A terminology $\mathcal{T}$ is *acyclic* if the transitive closure of $\prec_\mathcal{T}$ is irreflexive; otherwise $\mathcal{T}$ is *cyclic*. An acyclic terminology can be unfolded (i.e. the process of substituting concept names by their definitions stops).

In this paper we do not restrict terminologies to be acyclic. However, we have to take care of certain cycles. In our approach we want all conjunctions to be unfolded. That is, for any conjunction $A_1 \sqcap \cdots \sqcap A_m$ in $\mathcal{T}$, we substitute any A_i with $B_1 \sqcap \cdots \sqcap B_n$ if $A_i \equiv B_1 \sqcap \cdots \sqcap B_n \in \mathcal{T}$. To this end we need to handle the cycles along such concept equations. Formally, a terminology $\mathcal{T}$ *has unfoldable conjunctions* if it does not contain any concept equations $A_1 \equiv F_1, \ldots, A_n \equiv F_n$, where $F_1, \ldots, F_n$ are conjunctions of concept names such that $A_{i+1} \in \mathsf{sig}(F_i)$ for every $1 \leq i < n$, and $A_1 \in \mathsf{sig}(F_n)$. Any terminology can be rewritten such that it has unfoldable conjunctions without changing the logical consequences (cf. proof of Lemma 1 in [5]). We say that a concept name A is *conjunctive in* $\mathcal{T}$ iff there exist concept names $B_1, \ldots, B_n$, $n > 0$, such that $A \equiv B_1 \sqcap \ldots \sqcap B_n \in \mathcal{T}$; otherwise A is said to be *non-conjunctive in* $\mathcal{T}$. Note that after the unfolding of conjunctions (and removing of cycles) in a terminology $\mathcal{T}$ no concept name that appears as a conjunct is defined as a conjunction in $\mathcal{T}$.

To simplify the presentation we assume that terminologies do not contain trivial axioms of the form $A \equiv \top$ or $A \equiv B$, where A and B are concept names.

An $\mathcal{EL}$-terminology $\mathcal{T}$ is *normalised* if it consists of $\mathcal{EL}$-concept inclusions and equations of the following forms:

- $A \equiv \exists r.B$, $A \equiv \exists r.\top$, $A \equiv B_1 \sqcap \ldots \sqcap B_m$, and
- $A \sqsubseteq \exists r.B$, $A \sqsubseteq \exists r.\top$, $A \sqsubseteq B_1 \sqcap \ldots \sqcap B_n$,

where $m \geq 2$, $n \geq 1$, and A, B, B_i are concept names such that every conjunct B_i is non-conjunctive in $\mathcal{T}$.

2.3 Logical Difference

The logical difference between two TBoxes witnessed by concept inclusions over a signature Σ is defined as follows.

Definition 2. The Σ-*concept difference* between two $\mathcal{EL}$-TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ for a signature Σ is the set $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ of all $\mathcal{EL}$-concept inclusions α such that $\mathsf{sig}(\alpha) \subseteq \Sigma$, $\mathcal{T}_1 \models \alpha$, and $\mathcal{T}_2 \not\models \alpha$.

As the set $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ is infinite in general, we make use of the following "primitive witnesses" theorem from [5] that states that we only have to consider two specific types of concept differences.

THEOREM 1 (PRIMITIVE WITNESSES). *Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be $\mathcal{EL}$-terminologies and Σ a signature. If $\alpha \in \mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$, then either $C \sqsubseteq A$ or $A \sqsubseteq D$ is a member of $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$, where $A \in \mathsf{sig}(\alpha)$ is a concept name and C, D are $\mathcal{EL}$-concepts occurring in α.*

We define $\mathsf{cWtn}^{\mathsf{lhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ as the set of all concept names A from Σ such that there exists an $\mathcal{EL}_\Sigma$-concept C with $A \sqsubseteq C \in \mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$. Similarly, $\mathsf{cWtn}^{\mathsf{rhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ is the set of all concept names $A \in \Sigma$ such that there exists an $\mathcal{EL}_\Sigma$-concept C with $C \sqsubseteq A \in \mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$. The concept names in $\mathsf{cWtn}^{\mathsf{lhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ are called *left-hand side witnesses* and the concept names in $\mathsf{cWtn}^{\mathsf{rhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ *right-hand side witnesses*. Note that these sets are subsets of Σ, and by Theorem 1 their union is a finite and succinct representation of the set $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$, which is typically infinite.

Checking for the concept difference between two terminologies equals checking for the existence of left- and right-hand side witnesses. As a corollary of Theorem 1, we have that: $\mathsf{Diff}_\Sigma(\mathcal{T}_1,\mathcal{T}_2)=\emptyset$ iff $\mathsf{cWtn}^{\mathsf{lhs}}_\Sigma(\mathcal{T}_1,\mathcal{T}_2)=\emptyset$ and $\mathsf{cWtn}^{\mathsf{rhs}}_\Sigma(\mathcal{T}_1,\mathcal{T}_2)=\emptyset$.

2.4 Ontology Hypergraphs

Hypergraphs are a generalisation of graphs with many applications in computer science and discrete mathematics. In knowledge representation hypergraphs have been used implicitly to define reachability-based modules of ontologies [11], and explicitly to define locality-based modules [10]. In this paper we also make the notion of a hypergraph explicit by transforming terminologies into hypergraphs in order to be able to define simulations on the graphs.

A *directed hypergraph* is a tuple $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a non-empty set of *nodes* (or *vertices*), and $\mathcal{E}$ is a set of *directed hyperedges* of the form $e = (S, S')$, where $S, S' \subseteq \mathcal{V}$. We use hypergraphs to represent terminologies as follows.

Definition 3. For a normalised terminology $\mathcal{T}$ and a signature Σ, the *ontology hypergraph* $\mathcal{G}^\Sigma_\mathcal{T}$ of $\mathcal{T}$ for Σ is a directed hypergraph $\mathcal{G}^\Sigma_\mathcal{T} = (\mathcal{V}, \mathcal{E})$ defined as follows:

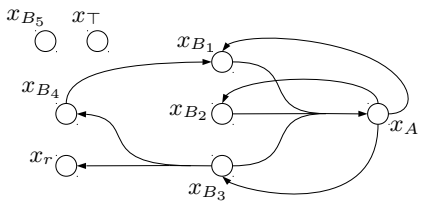
$$\begin{aligned}\mathcal{V} = {} & \{\, x_A \mid A \in \mathsf{N_C} \cap (\Sigma \cup \mathsf{sig}(\mathcal{T})) \,\} \\ & \cup \{\, x_r \mid r \in \mathsf{N_R} \cap (\Sigma \cup \mathsf{sig}(\mathcal{T})) \,\} \\ & \cup \{x_\top\}\end{aligned}$$

and

$$\begin{aligned}\mathcal{E} = {} & \{\, (\{x_A\}, \{x_{B_i}\}) \mid A \sqsubseteq B_1 \sqcap \ldots \sqcap B_n \in \mathcal{T},\ 1 \leq i \leq n \,\} \\ & \cup \{\, (\{x_A\}, \{x_{B_i}\}) \mid A \equiv B_1 \sqcap \ldots \sqcap B_n \sqsubseteq \mathcal{T},\ 1 \leq i \leq n \,\} \\ & \cup \{\, (\{x_A\}, \{x_r, x_Y\}) \mid A \sqsubseteq \exists r.Y \in \mathcal{T},\ Y \in \mathsf{N_C} \cup \{\top\} \,\} \\ & \cup \{\, (\{x_A\}, \{x_r, x_Y\}) \mid A \equiv \exists r.Y \in \mathcal{T},\ Y \in \mathsf{N_C} \cup \{\top\} \,\} \\ & \cup \{\, (\{x_r, x_Y\}, \{x_A\}) \mid A \equiv \exists r.Y \in \mathcal{T},\ Y \in \mathsf{N_C} \cup \{\top\} \,\} \\ & \cup \{\, (\{x_{B_1}, \ldots, x_{B_n}\}, \{x_A\}) \mid A \equiv B_1 \sqcap \ldots \sqcap B_n \in \mathcal{T} \,\}\end{aligned}$$

An ontology hypergraph $\mathcal{G}_{\mathcal{T}}^{\Sigma}$ contains a node for $\top$ and for every role and concept name in Σ or $\mathcal{T}$. Hyperedges in $\mathcal{G}_{\mathcal{T}}^{\Sigma}$ represent axioms in $\mathcal{T}$. Every hyperedge is directed and can be understood as an implication, i.e., $(\{x_A\}, \{x_B\})$ represents $\mathcal{T} \models A \sqsubseteq B$. The complex hyperedges are of the form $(\{x_A\}, \{x_r, x_B\})$ and $(\{x_r, x_B\}, \{x_A\})$ representing $\mathcal{T} \models A \sqsubseteq \exists r.B$ and $\mathcal{T} \models \exists r.B \sqsubseteq A$, and of the form $(\{x_{B_1}, ..., x_{B_n}\}, \{x_A\})$ standing for $\mathcal{T} \models B_1 \sqcap \ldots \sqcap B_n \sqsubseteq A$. Note that due to the normalisation of $\mathcal{T}$, conjunctions always have more than one conjunct (i.e. $n \geq 2$).

Example 1. Let $\mathcal{T} = \{A \equiv B_1 \sqcap B_2 \sqcap B_3,\ B_3 \sqsubseteq \exists r.B_4, B_4 \sqsubseteq B_1\}$ and $\Sigma = \{B_5\}$. Then the ontology hypergraph $\mathcal{G}_{\mathcal{T}}^{\Sigma}$ of $\mathcal{T}$ for Σ can be depicted as follows:



3. LOGICAL DIFFERENCE USING HYPERGRAPHS

Our approach for detecting logical differences wrt. Σ is based on finding appropriate simulations between the hypergraphs $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ and $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ such that every node x_A in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ with $A \in \Sigma$ is simulated by the node x_A in $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$. It is well known that the existence of a simulation between two graph structures can be used to characterise some notion of equivalence between the graphs [4], for example reachability. In this paper we aim to capture logical entailment wrt. a signature by defining the simulation relations appropriately.

We first introduce an auxiliary relation $\rightarrow_{\mathcal{T}}$ over the nodes of the ontology hypergraph $\mathcal{G}_{\mathcal{T}}^{\Sigma}$ of the terminology $\mathcal{T}$. The relation $\rightarrow_{\mathcal{T}}$ is a *special* reachability notion in $\mathcal{G}_{\mathcal{T}}^{\Sigma}$ that mimics reasoning wrt. $\mathcal{T}$. The definition of $\rightarrow_{\mathcal{T}}$ is related to the completion algorithm for classification in $\mathcal{EL}$ [1] and OWL 2 QL [8]. Afterwards we define two types of simulations between the hypergraphs of two terminologies $\mathcal{T}_1$ and $\mathcal{T}_2$, one type of simulation for each type of witness.

Definition 4. Let $\mathcal{G}_{\mathcal{T}}^{\Sigma} = (\mathcal{V}, \mathcal{E})$ be the ontology hypergraph of a normalised terminology $\mathcal{T}$ for a signature Σ. The relation $\rightarrow_{\mathcal{T}} \subseteq \mathcal{V}(1) \times \mathcal{V}(2)$ is inductively defined as follows, where $\mathcal{V}(k) = \{\, S \subseteq \mathcal{V} \mid 0 < |S| \leq k \,\}$:

(i) $\{x\} \rightarrow_{\mathcal{T}} \{x\}$ for every $x \in \mathcal{V}$;

(ii) $\{x\} \rightarrow_{\mathcal{T}} \{z\}$ if $\{x\} \rightarrow_{\mathcal{T}} \{y\}$, $(\{y\}, \{z\}) \in \mathcal{E}$;

(iii) $\{x\} \rightarrow_{\mathcal{T}} \{x_r, z\}$ if $\{x\} \rightarrow_{\mathcal{T}} \{y\}$, $(\{y\}, \{x_r, z\}) \in \mathcal{E}$;

(iv) $\{x\} \rightarrow_{\mathcal{T}} \{z\}$ if $\{x\} \rightarrow_{\mathcal{T}} \{x_r, y\}$, $\{y\} \rightarrow_{\mathcal{T}} \{y'\}$, and $(\{x_r, y'\}, \{z\}) \in \mathcal{E}$;

(v) $\{x\} \rightarrow_{\mathcal{T}} \{z\}$ if $\{x\} \rightarrow_{\mathcal{T}} \{x_r, y\}$, $(\{x_r, x_\top\}, \{z\}) \in \mathcal{E}$;

(vi) $\{x\} \rightarrow_{\mathcal{T}} \{z\}$ if $\{x\} \rightarrow_{\mathcal{T}} \{y_i\}$ for all $i \in \{1, \ldots, n\}$, $(\{y_1, \ldots, y_n\}, \{z\}) \in \mathcal{E}$.

Note that the relation $\rightarrow_{\mathcal{T}}$ associates nodes x_A that represent concept names A either with nodes x_B that stand for concept names B or with pairs of nodes $\{x_r, z\}$ representing concepts of the form $\exists r.A$ or $\exists r.\top$. The binary relation $\rightarrow_{\mathcal{T}}$ is reflexive and transitive on single nodes by Conditions (i) and (ii). Moreover, in Condition (vi) transitivity of $\rightarrow_{\mathcal{T}}$ is extended to hyperedges with complex left-hand sides, representing axioms of the form $A \equiv B_1 \sqcap \ldots \sqcap B_n$. The other conditions handle pairs of nodes. Condition (iii) states that any indirectly reachable pair $\{x_r, z\}$ via an intermediate node is also directly reachable via $\rightarrow_{\mathcal{T}}$, while Condition (iv) ensures the same property for indirectly reachable nodes via intermediate pairs. Condition (v) is a special case of (iv) for handling pairs involving $\top$ as ontology hypergraphs for normalised terminologies $\mathcal{T}$ do not contain hyperedges from nodes x_A representing concept names to $x_\top$ representing $\top$ ($\mathcal{T}$ does not contain any axioms of the form $A \sqsubseteq \top$ or $A \equiv \top$).

It can be readily seen that the relation $\rightarrow_{\mathcal{T}}$ can be computed in polynomial time.

We emphasise here that the relation $\rightarrow_{\mathcal{T}}$ does *not* coincide with the usual reachability notion in a hypergraph. The following example shows that $\rightarrow_{\mathcal{T}}$ connects reachable nodes but not all reachable nodes are connected via $\rightarrow_{\mathcal{T}}$. This means that the usual reachability relation does not correctly mimic logical consequences entailed by $\mathcal{T}$.

Example 2. Let $\mathcal{T} = \{A \sqsubseteq \exists r.B',\ \exists r.B' \sqsubseteq B,\ \exists r.B \sqsubseteq A'\}$. It holds that $\{x_A\} \rightarrow_{\mathcal{T}} \{x_B\}$, i.e. $\mathcal{T} \models A \sqsubseteq B$, and the node x_B is reachable from x_A (in terms of standard graph reachability). However, $x_{A'}$ is also reachable from x_A whereas $\{x_A\} \not\rightarrow_{\mathcal{T}} \{x_{A'}\}$ and $\mathcal{T} \not\models A \sqsubseteq A'$.

The notion of reachability induced by the relation $\rightarrow_{\mathcal{T}}$ can be characterised in terms of entailment.

Lemma 2. Let $\mathcal{G}_{\mathcal{T}}^{\Sigma} = (\mathcal{V}, \mathcal{E})$ be the ontology hypergraph of a normalised terminology $\mathcal{T}$ for a signature Σ. Then we have for every $A, B, r \in \Sigma \cup \mathsf{sig}(\mathcal{T})$:

(i) $\mathcal{T} \models A \sqsubseteq B$ iff $\{x_A\} \rightarrow_{\mathcal{T}} \{x_B\}$;

(ii) $\mathcal{T} \models A \sqsubseteq \exists r.B$ iff $\{x_A\} \rightarrow_{\mathcal{T}} \{x_r, x_{B'}\}$ and $\{x_{B'}\} \rightarrow_{\mathcal{T}} \{x_B\}$ for some $B' \in \Sigma \cup \mathsf{sig}(\mathcal{T})$;

(iii) $\mathcal{T} \models A \sqsubseteq \exists r.\top$ iff $\{x_A\} \rightarrow_{\mathcal{T}} \{x_r, x_Y\}$ for some $Y \in \Sigma \cup \mathsf{sig}(\mathcal{T}) \cup \{\top\}$.

As described above, we want to check for every concept name $A \in \Sigma$ whether A belongs to $\mathsf{cWtn}_{\Sigma}^{\mathsf{lhs}}(\mathcal{T}_1, \mathcal{T}_2)$ or to $\mathsf{cWtn}_{\Sigma}^{\mathsf{rhs}}(\mathcal{T}_1, \mathcal{T}_2)$. For the former, we check for the existence of a *forward simulation*, and for the latter, for the existence of a *backward simulation* between the ontology hypergraphs $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ and $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$. We define the simulations in the following subsections.

3.1 Forward Simulation

Based on the relation $\to_{\mathcal{T}}$ we can now give the definition of the forward simulation, which connects nodes in $\mathcal{G}^{\Sigma}_{\mathcal{T}_1}$ with nodes in $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$ that are reachable via $\to_{\mathcal{T}_1}$ and $\to_{\mathcal{T}_2}$, respectively.

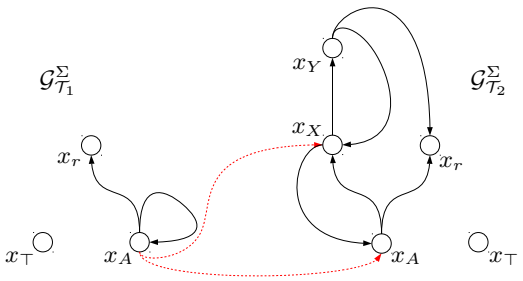
Definition 5. Let $\mathcal{G}^{\Sigma}_{\mathcal{T}_1} = (\mathcal{V}_1, \mathcal{E}_1)$, $\mathcal{G}^{\Sigma}_{\mathcal{T}_2} = (\mathcal{V}_2, \mathcal{E}_2)$ be ontology hypergraphs of two normalised terminologies $\mathcal{T}_1$ and $\mathcal{T}_2$ for a signature Σ. A relation $\hookrightarrow^{f}_{\Sigma} \subseteq \mathcal{V}_1 \times \mathcal{V}_2$ is a *forward* Σ-simulation between $\mathcal{G}^{\Sigma}_{\mathcal{T}_1}$ and $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$ if the following conditions hold:

- (i_f) if $x_A \hookrightarrow^{f}_{\Sigma} x_{A'}$, then for every $B \in \Sigma$ with $\{x_A\} \to_{\mathcal{T}_1} \{x_B\}$ it holds that $\{x_{A'}\} \to_{\mathcal{T}_2} \{x_B\}$;
- (ii_f) if $x_A \hookrightarrow^{f}_{\Sigma} x_{A'}$, then for every $r \in \Sigma$ such that $\{x_A\} \to_{\mathcal{T}_1} \{x_r, x_X\}$ there is a $x_{X'} \in \mathcal{V}_2$ such that $\{x_{A'}\} \to_{\mathcal{T}_2} \{x_r, x_{X'}\}$ and $x_X \hookrightarrow^{f}_{\Sigma} x_{X'}$.

We write $\mathcal{G}^{\Sigma}_{\mathcal{T}_1} \hookrightarrow^{f}_{\Sigma} \mathcal{G}^{\Sigma}_{\mathcal{T}_2}$ iff there exists a forward Σ-simulation $\hookrightarrow^{f}_{\Sigma} \subseteq \mathcal{V}_1 \times \mathcal{V}_2$ such that $(x_A, x_A) \in \hookrightarrow^{f}_{\Sigma}$ for every $A \in \Sigma$.

For a node x_A in $\mathcal{G}^{\Sigma}_{\mathcal{T}_1}$ to be forward simulated by $x_{A'}$ in $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$, Condition (i_f) enforces that every Σ-concept name B that is entailed by A in $\mathcal{T}_1$ must also be entailed by A' in $\mathcal{T}_2$. Condition (ii_f) ensures a similar requirement for concepts of the form $\exists r.X$ with $X \in \mathsf{sig}(\mathcal{T}_1) \cup \{\top\}$ such that $\mathcal{T}_1 \models A \sqsubseteq \exists r.X$ while propagating the simulation to the successor node x_X.

Example 3. Let $\mathcal{T}_1 = \{A \sqsubseteq \exists r.A\}$, $\mathcal{T}_2 = \{A \sqsubseteq \exists r.X,\ X \sqsubseteq A \sqcap Y,\ Y \sqsubseteq \exists r.X\}$, and $\Sigma = \{A, r\}$. Then one can see that $\mathsf{Diff}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$. Furthermore, wrt. $\mathcal{G}^{\Sigma}_{\mathcal{T}_1}$ it only holds that $\{x_A\} \to_{\mathcal{T}_1} \{x_A\}$, $\{x_A\} \to_{\mathcal{T}_1} \{x_r, x_A\}$. Regarding $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$, we have $\{x_A\} \to_{\mathcal{T}_2} \{x_A\}$, $\{x_A\} \to_{\mathcal{T}_2} \{x_r, x_X\}$, $\{x_X\} \to_{\mathcal{T}_2} \{x_A\}$, $\{x_X\} \to_{\mathcal{T}_2} \{x_r, x_X\}$. Hence, one can see that $S = \{(x_A, x_A), (x_A, x_X)\}$ is a forward Σ-simulation between $\mathcal{G}^{\Sigma}_{\mathcal{T}_1}$ and $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$ with $(x_A, x_A) \in S$. A graphical representation of the ontology hypergraphs $\mathcal{G}^{\Sigma}_{\mathcal{T}_1}$, $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$ and of the simulation S can be found below.



Example 4. Let $\mathcal{T}_1 = \{A \sqsubseteq \exists r.X,\ X \sqsubseteq A \sqcap B\}$, $\mathcal{T}_2 = \{A \sqsubseteq X \sqcap Y,\ X \sqsubseteq \exists r.A,\ Y \sqsubseteq \exists r.B\}$, and $\Sigma = \{A, B, r\}$. Then, for instance, $A \sqsubseteq \exists r.(A \sqcap B) \in \mathsf{Diff}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$. It holds that $\{x_A\} \to_{\mathcal{T}_1} \{x_r, x_X\}$, $\{x_X\} \to_{\mathcal{T}_1} \{x_A\}$, $\{x_X\} \to_{\mathcal{T}_1} \{x_B\}$, $\{x_A\} \to_{\mathcal{T}_2} \{x_r, x_A\}$, $\{x_A\} \to_{\mathcal{T}_2} \{x_r, x_B\}$. However, for $x = x_A$ or $x = x_B$ it does not hold that $\{x\} \to_{\mathcal{T}_2} \{x_A\}$ *and* $\{x\} \to_{\mathcal{T}_2} \{x_B\}$, i.e. the node x_X in $\mathcal{G}^{\Sigma}_{\mathcal{T}_1}$ cannot be simulated by x_A or x_B in $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$ as Condition (i_f) cannot be satisfied. Thus, one can see that there cannot exist a forward Σ-simulation S between $\mathcal{G}^{\Sigma}_{\mathcal{T}_1}$ and $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$ with $(x_A, x_A) \in S$.

We now prove that the existence of a forward simulation between a node x_{A_1} in $\mathcal{G}_{\mathcal{T}_1}$ and a node x_{A_2} in $\mathcal{G}_{\mathcal{T}_2}$ exactly captures the property that $\mathcal{T}_1 \models A_1 \sqsubseteq C$ entails that $\mathcal{T}_2 \models A_2 \sqsubseteq C$ for every Σ-concept C.

Lemma 3. Let $\mathcal{T}_1, \mathcal{T}_2$ be normalised terminologies, and let Σ be a signature such that $\mathcal{G}_{\mathcal{T}_1} \hookrightarrow^{f}_{\Sigma} \mathcal{G}_{\mathcal{T}_2}$. Then for every $\mathcal{EL}_{\Sigma}$-concept C and for every $(x_{A_1}, x_{A_2}) \in \hookrightarrow^{f}_{\Sigma}$ with $\mathcal{T}_1 \models A_1 \sqsubseteq C$ it holds that $\mathcal{T}_2 \models A_2 \sqsubseteq C$.

Lemma 4. Let $\mathcal{T}_1, \mathcal{T}_2$ be normalised terminologies, and let Σ be a signature such that $\mathsf{cWtn}^{\mathsf{lhs}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$. Then $\mathcal{G}_{\mathcal{T}_1} \hookrightarrow^{f}_{\Sigma} \mathcal{G}_{\mathcal{T}_2}$.

We obtain Theorem 2 as a consequence of the previous two lemmas.

THEOREM 2. *Let $\mathcal{T}_1, \mathcal{T}_2$ be normalised terminologies, and let Σ be a signature. Then $\mathsf{cWtn}^{\mathsf{lhs}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$ iff $\mathcal{G}_{\mathcal{T}_1} \hookrightarrow^{f}_{\Sigma} \mathcal{G}_{\mathcal{T}_2}$.*

3.2 Backward Simulation

We now turn to right-hand side witnesses, i.e. we want to devise an algorithm that checks whether $\mathsf{cWtn}^{\mathsf{rhs}}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$. Analogously as for the left-hand side witnesses, we introduce a *backward* simulation which has the property that a node x_{A_1} in $\mathcal{G}^{\Sigma}_{\mathcal{T}_1}$ is simulated by a node x_{A_2} in $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$ iff $\mathcal{T}_1 \models C \sqsubseteq A_1$ entails $\mathcal{T}_2 \models C \sqsubseteq A_2$ for every Σ-concept C. Intuitively, the hypergraph has to be traversed backwards to identify all essential concepts C for which $\mathcal{T}_1 \models C \sqsubseteq A_1$. In particular, concept names A_1 for which there does not exist an $\mathcal{EL}_{\Sigma}$-concept C with $\mathcal{T}_1 \models C \sqsubseteq A_1$ do not have to be simulated by a node in $\mathcal{G}^{\Sigma}_{\mathcal{T}_2}$ since such concept names cannot become right-hand side witnesses. We identify such concept names A_1 by checking whether the node x_{A_1} is Σ-*entailed* in the following sense.

Definition 6. Let $\mathcal{G}^{\Sigma}_{\mathcal{T}} = (\mathcal{V}, \mathcal{E})$ be the ontology hypergraph of a normalised terminology $\mathcal{T}$ for a signature Σ. Moreover, let $\mathcal{V}_{\Sigma} \subseteq \mathcal{V}$ be the smallest set of nodes defined inductively as follows:

- (i) $x_\top \in \mathcal{V}_{\Sigma}$;
- (ii) if $x_A \in \mathcal{V}$ such that there exists $B \in \Sigma$ with $\{x_B\} \to_{\mathcal{T}} \{x_A\}$, then $x_A \in \mathcal{V}_{\Sigma}$;
- (iii) if $x_B \in \mathcal{V}_{\Sigma}$ with $B \in \mathsf{N_C} \cup \{\top\}$, $(\{x_B, x_r\}, \{x_A\}) \in \mathcal{E}$, and $r \in \Sigma$, then $x_A \in \mathcal{V}_{\Sigma}$;
- (iv) if $x_{B_1}, \ldots, x_{B_n} \in \mathcal{V}_{\Sigma}$ with $(\{x_{B_1}, \ldots, x_{B_n}\}, \{x_A\}) \in \mathcal{E}$, then $x_A \in \mathcal{V}_{\Sigma}$.

We then say that a node $x \in \mathcal{V}$ is Σ-*entailed in* $\mathcal{G}^{\Sigma}_{\mathcal{T}}$ iff $x \in \mathcal{V}_{\Sigma}$.

The node $x_\top$ is always Σ-entailed for every signature Σ. A node x is Σ-entailed if it is reachable via $\to_{\mathcal{T}}$ from a node x_B with $B \in \Sigma$, or if its direct predecessors in the ontology hypergraph are Σ-entailed. In particular, every node x_A with $A \in \Sigma$ is Σ-entailed.

Example 5. Let $\mathcal{T} = \{A \equiv \exists r.X,\ X \equiv B_1 \sqcap B_2\}$. For $\Sigma_1 = \{B_1, B_2, r\}$, all the nodes are Σ_1-entailed in $\mathcal{G}_{\mathcal{T}}^{\Sigma_1}$. However, for $\Sigma_2 = \{B_1, B_2\}$ only the nodes x_{B_1}, x_{B_2} x_X, and $x_\top$ are Σ_2-entailed in $\mathcal{G}_{\mathcal{T}}^{\Sigma_2}$, whereas for $\Sigma_3 = \{B_1, r\}$ only the node $x_\top$ is Σ_3-entailed in $\mathcal{G}_{\mathcal{T}}^{\Sigma_3}$. Note that $\mathcal{T} \models C \sqsubseteq A$ holds for $C = \exists r.(B_1 \sqcap B_2)$ and $\mathsf{sig}(C) \subseteq \Sigma_1$ but $\mathsf{sig}(C) \not\subseteq \Sigma_2$ and $\mathsf{sig}(C) \not\subseteq \Sigma_3$.

Lemma 5. Let $\mathcal{G}_{\mathcal{T}}^{\Sigma} = (\mathcal{V}, \mathcal{E})$ be the ontology hypergraph of a normalised terminology $\mathcal{T}$ for a signature Σ, and let $x_A \in \mathcal{V}$. Then the node x_A is Σ-entailed in $\mathcal{G}_{\mathcal{T}}^{\Sigma}$ iff there exists an $\mathcal{EL}_\Sigma$-concept C such that $\mathcal{T} \models C \sqsubseteq A$.

To compute all the nodes in a given graph $\mathcal{G}_{\mathcal{T}}$ that are Σ-entailed, we can proceed as follows. In a first step identify all the nodes x that fulfill conditions (i) and (ii) by using the relation $\rightarrow_{\mathcal{T}}$. Subsequently, propagate the Σ-entailed status to other nodes using conditions (iii) and (iv). It can be readily seen that these computation steps can be performed in polynomial time.

Before we can give the definition of the backward simulation, we have to introduce the following notion: we associate with every node x_A in a hypergraph $\mathcal{G}_{\mathcal{T}}$ a set of concept names non-conj(x_A) which are "essential" to entail A in $\mathcal{T}$ (also see [5] for a similar notion).

Definition 7. Let $\mathcal{G}_{\mathcal{T}}^{\Sigma} = (\mathcal{V}, \mathcal{E})$ be an ontology hypergraph. For $x_A \in \mathcal{V}$, let non-conj(x_A) be defined as follows

- if $(\{x_{B_1}, \ldots, x_{B_n}\}, \{x_A\}) \in E$, let
$$\text{non-conj}_{\mathcal{T}}(x_A) = \{x_{B_1}, \ldots, x_{B_n}\};$$
- otherwise, let non-conj$_{\mathcal{T}}(x_A) = \{x_A\}$.

For a graph $\mathcal{G}_{\mathcal{T}}^{\Sigma} = (\mathcal{V}, \mathcal{E})$ we have $(\{x_{B_1}, \ldots, x_{B_n}\}, \{x_A\}) \in \mathcal{E}$ iff $A \equiv B_1 \sqcap \ldots \sqcap B_n \in \mathcal{T}$. Hence, it holds for every $\mathcal{EL}_\Sigma$-concept C that $\mathcal{T} \models C \sqsubseteq A$ iff $\mathcal{T} \models C \sqsubseteq X$ for every $X \in \{\, X \mid x_X \in \text{non-conj}_{\mathcal{T}}(x_A) \,\}$.

We can now give the definition of a *backward simulation*.

Definition 8. Let $\mathcal{G}_{\mathcal{T}_1}^{\Sigma} = (\mathcal{V}_1, \mathcal{E}_1)$, $\mathcal{G}_{\mathcal{T}_2}^{\Sigma} = (\mathcal{V}_2, \mathcal{E}_2)$ be the ontology hypergraphs of the normalised terminologies $\mathcal{T}_1$ and $\mathcal{T}_2$ for a signature Σ. A relation $\hookrightarrow_\Sigma^b \subseteq \mathcal{V}_1 \times \mathcal{V}_2$ is a *backward* Σ-simulation between $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ and $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ if the following conditions hold:

(i_b) if $x_A \hookrightarrow_\Sigma^b x_{A'}$, then for every $B \in \Sigma$ with $\{x_B\} \rightarrow_{\mathcal{T}_1} \{x_A\}$ it holds that $\{x_B\} \rightarrow_{\mathcal{T}_2} \{x'_A\}$;

(ii_b) if $x_A \hookrightarrow_\Sigma^b x_{A'}$ and $(\{x_X, x_r\}, \{x_A\}) \in \mathcal{E}_1$ such that $r \in \Sigma$ and x_X is Σ-entailed in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$, then for every $x_{B'_i} \in$ non-conj$_{\mathcal{T}_2}(x_{A'})$ there exists $(\{x_{X'_i}, x_r\}, \{x_{B'_i}\}) \in \mathcal{E}_2$ such that $x_X \hookrightarrow_\Sigma^b x_{X'_i}$;

(iii_b) if $x_A \hookrightarrow_\Sigma^b x_{A'}$ and $(\{x_{B_1}, \ldots x_{B_n}\}, \{x_A\}) \in \mathcal{E}_1$ where x_{B_i} are Σ-entailed in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ for every $1 \leq i \leq n$, then for every $x' \in$ non-conj$_{\mathcal{T}_2}(x_A)$ there exists an $x \in$ non-conj$_{\mathcal{T}_1}(x_A)$ with $x \hookrightarrow_\Sigma^b x'$.

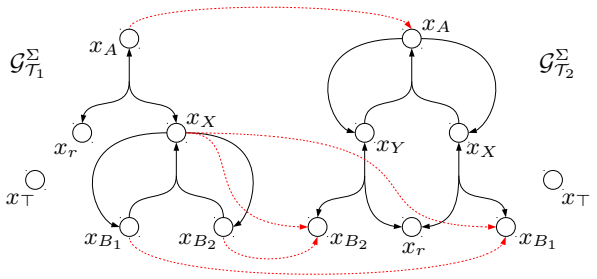
In the following, we write $\mathcal{G}_{\mathcal{T}_1}^{\Sigma} \hookrightarrow_\Sigma^b \mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ iff there exists a backward Σ-simulation $\hookrightarrow_\Sigma^b \subseteq \mathcal{V}_1 \times \mathcal{V}_2$ with $(x_A, x_A) \in \hookrightarrow_\Sigma^b$ for every $A \in \Sigma$.

For a node x_A in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ to be backward simulated by $x_{A'}$ in $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$, Conditions ($i_b$) and ($ii_b$) are the equivalent of the Conditions (i_f) and (ii_f), respectively, for forward simulations. Condition (iii_b) handles axioms of the form $A \equiv B_1 \sqcap \ldots \sqcap B_n$ in $\mathcal{T}_1$. Note that we quantify over the conjuncts of A' in $\mathcal{T}_2$ since, intuitively speaking, fewer conjuncts suffice to preserve logical entailments. Take, for instance, the two normalised terminologies $\mathcal{T}_1 = \{A \equiv B_1 \sqcap B_2\}$, $\mathcal{T}_2 = \{A \sqsubseteq B_1 \sqcap B_2,\ B_1 \sqsubseteq A\}$ and the signature $\Sigma = \{A, B_1, B_2\}$; then $\mathsf{cWtn}_\Sigma^{\mathsf{rhs}}(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$ and, in particular, $\mathcal{T}_2 \models B_1 \sqcap B_2 \sqsubseteq A$ holds as well.

Example 6. Let $\mathcal{T}_1 = \{A \equiv \exists r.X,\ X \equiv B_1 \sqcap B_2\}$, $\mathcal{T}_2 = \{A \equiv X \sqcap Y, X \equiv \exists r.B_1, Y \equiv \exists r.B_2\}$, and $\Sigma = \{A, B_1, B_2, r\}$. First we observe that the nodes x_{B_1}, x_{B_2}, x_X, and x_A are Σ-entailed in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$. As only $\{x_{B_i}\} \rightarrow_{\mathcal{T}_1} \{x_{B_i}\}$ for $i \in \{1, 2\}$, one can see that the node x_{B_i} in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ can be simulated by the node x_{B_i} in $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ for $i \in \{1, 2\}$. Due to non-conj$_{\mathcal{T}_2}(x_{B_i}) = \{x_{B_i}\}$ for $i \in \{1, 2\}$ and non-conj$_{\mathcal{T}_1}(x_X) = \{x_{B_1}, x_{B_2}\}$, we can infer that the node x_X in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ can be simulated both by x_{B_1} and x_{B_2} in $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ (there does not exist $X' \in \Sigma$ with $\{x_{X'}\} \rightarrow_{\mathcal{T}_1} \{x_X\}$). Finally, as non-conj$_{\mathcal{T}_2}(x_A) = \{x_X, x_Y\}$, we conclude that the node x_A in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ can be simulated by x_A in $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ due to Condition (ii_b) (Condition (i_b) is trivially satisfied). Overall,

$$S = \{(x_A, x_A), (x_X, x_{B_1}), (x_X, x_{B_2}), (x_{B_1}, x_{B_1}), (x_{B_2}, x_{B_2})\}$$

is a backward Σ-simulation between $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ and $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ such that $(Z, Z) \in S$ for every $Z \in \mathsf{N_C} \cap \Sigma$. A graphical representation of the ontology hypergraphs $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$, $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ and of the simulation S can be found below.



Example 7. Let $\mathcal{T}_1 = \{A \equiv B_1 \sqcap B_2\}$, $\mathcal{T}_2 = \{A \equiv B_1 \sqcap B'\}$, and $\Sigma = \{A, B_1, B_2\}$. First we observe that there does not exist a concept name $Z \in \Sigma$ with $\{x_Z\} \rightarrow_{\mathcal{T}_2} \{x_{B'}\}$, i.e. the nodes x_{B_1}, x_{B_2} in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ cannot be simulated by $x_{B'}$ in $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ as Condition (i_b) would be violated. Hence, as non-conj$_{\mathcal{T}_1}(x_A) = \{x_{B_1}, x_{B_2}\}$ and as non-conj$_{\mathcal{T}_2}(x_A) = \{x_{B_1}, x_{B'}\}$, we can conclude that there cannot exist a backward Σ-simulation such that x_A in $\mathcal{G}_{\mathcal{T}_1}^{\Sigma}$ is simulated by x_A in $\mathcal{G}_{\mathcal{T}_2}^{\Sigma}$ as Condition (iii_b) cannot be fulfilled.

We can now establish the correctness and completeness properties regarding backward simulations.

Lemma 6. Let $\mathcal{T}_1, \mathcal{T}_2$ be normalised terminologies, and let Σ be a signature such that $\mathcal{G}_{\mathcal{T}_1} \hookrightarrow^b_\Sigma \mathcal{G}_{\mathcal{T}_2}$. Then for every $\mathcal{EL}_\Sigma$-concept C and for every $(x_{A_1}, x_{A_2}) \in \hookrightarrow^b_\Sigma$ with $\mathcal{T}_1 \models C \sqsubseteq A_1$ it holds that $\mathcal{T}_2 \models C \sqsubseteq A_2$.

Lemma 7. Let $\mathcal{T}_1, \mathcal{T}_2$ be normalised terminologies, and let Σ be a signature such that $\mathsf{cWtn}^{\mathsf{rhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$. Then $\mathcal{G}_{\mathcal{T}_1} \hookrightarrow^b_\Sigma \mathcal{G}_{\mathcal{T}_2}$.

We obtain Theorem 3 as a consequence of the previous two lemmas.

THEOREM 3. *Let $\mathcal{T}_1, \mathcal{T}_2$ be normalised terminologies, and let Σ be a signature with $A \in \Sigma$. Then $\mathsf{cWtn}^{\mathsf{rhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$ iff $\mathcal{G}_{\mathcal{T}_1} \hookrightarrow^b_\Sigma \mathcal{G}_{\mathcal{T}_2}$.*

3.3 Computational Complexity

Given two hypergraphs $\mathcal{G}^\Sigma_{\mathcal{T}_1} = (\mathcal{V}_1, \mathcal{E}_1)$ and $\mathcal{G}^\Sigma_{\mathcal{T}_2} = (\mathcal{V}_2, \mathcal{E}_2)$, one can proceed as follows to check whether $\mathcal{G}^\Sigma_{\mathcal{T}_1} \hookrightarrow^f_\Sigma \mathcal{G}^\Sigma_{\mathcal{T}_2}$ holds. First, let $S^f_0 \subseteq \mathcal{V}_1 \times \mathcal{V}_2$ be the set of all the pairs that fulfill Conditions (i_f). Subsequently, iterate over the elements contained in the set S^f_i and remove those pairs which do not satisfy Conditions (ii_f) to obtain the set S^f_{i+1}. Eventually we will have $S^f_j = S^f_{j+1}$ for some index j and one can conclude that $\mathcal{G}^\Sigma_{\mathcal{T}_1} \hookrightarrow^f_\Sigma \mathcal{G}^\Sigma_{\mathcal{T}_2}$ holds iff $(x_A, x_A) \in S^f_j$ for every $A \in \Sigma$.

It is easy to see that the simulation Conditions (i_f) and (ii_f) can be checked in polynomial time. Thus, as the procedure described above terminates in at most $|\mathcal{V}_1 \times \mathcal{V}_2|$ iterations, we can infer that it can be checked in polynomial time whether $\mathcal{G}_{\mathcal{T}_1} \hookrightarrow^f_\Sigma \mathcal{G}_{\mathcal{T}_2}$ holds.

Similar arguments show that the existence of a backward Σ-simulation can be checked in polynomial time as well, which gives us the following result.

THEOREM 4. *Let $\mathcal{G}^\Sigma_{\mathcal{T}_1} = (\mathcal{V}_1, \mathcal{E}_1)$, $\mathcal{G}^\Sigma_{\mathcal{T}_2} = (\mathcal{V}_2, \mathcal{E}_2)$ be ontology hypergraphs of two normalised terminologies $\mathcal{T}_1$ and $\mathcal{T}_2$ for a signature Σ. Then it can be checked in polynomial time whether $\mathcal{G}^\Sigma_{\mathcal{T}_1} \hookrightarrow^f_\Sigma \mathcal{G}^\Sigma_{\mathcal{T}_2}$ and $\mathcal{G}^\Sigma_{\mathcal{T}_1} \hookrightarrow^b_\Sigma \mathcal{G}^\Sigma_{\mathcal{T}_2}$ holds.*

Note that in a practical implementation it would not be required to take the complete ontology graphs $\mathcal{G}^\Sigma_{\mathcal{T}_1}$ and $\mathcal{G}^\Sigma_{\mathcal{T}_2}$ into account if one wants to check whether a concept name $A \in \Sigma$ is a difference witness. It is sufficient to consider the subgraph only which is induced by the $\rightarrow_{\mathcal{T}_1}$ and $\rightarrow_{\mathcal{T}_2}$ either in the "forward" or "backward" direction depending on the type of witnesses that should be computed. For a typical (practical) terminology $\mathcal{T}$, $S \rightarrow_\mathcal{T} S'$ only holds for relatively few sets of nodes S, S', which suggests that the number of nodes that have to be considered for a simulation check should remain fairly small as well.

3.4 Computing Difference Examples

So far we have focused on finding difference witnesses, i.e. concept names A belonging either to the set $\mathsf{cWtn}^{\mathsf{lhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ or the set $\mathsf{cWtn}^{\mathsf{rhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$, which is sufficient to decide the existence of a logical difference between $\mathcal{T}_1$ and $\mathcal{T}_2$. However, in practical applications of logical difference it can be helpful for users to have a concrete concept inclusion $C \sqsubseteq A$ or $A \sqsubseteq D$ in $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ that corresponds to a witness A. We now sketch how to read such concept inclusions directly off a hypergraph using Example 7.

Recall that x_{B_1}, x_{B_2} in $\mathcal{G}^\Sigma_{\mathcal{T}_1}$ cannot be simulated by $x_{B'}$ in $\mathcal{G}^\Sigma_{\mathcal{T}_2}$ as $\mathcal{T}_2 \not\models B_1 \sqsubseteq B'$ and $\mathcal{T}_2 \not\models B_2 \sqsubseteq B'$, i.e. for the Σ-concept $C = B_1 \sqcap B_2$ it holds that $\mathcal{T}_1 \models C \sqsubseteq B_1 \sqcap B_2$, but $\mathcal{T}_2 \not\models C \sqsubseteq B_1 \sqcap B'$. Hence, we have $\mathcal{T}_1 \models C \sqsubseteq A$ but $\mathcal{T}_2 \not\models C \sqsubseteq A$.

In general, if a node x_A in $\mathcal{G}^\Sigma_{\mathcal{T}_1}$ cannot be simulated by x_A in $\mathcal{G}^\Sigma_{\mathcal{T}_2}$, there exists a node x in $\mathcal{G}^\Sigma_{\mathcal{T}_2}$ which is the main cause for the failure to find a simulation ($x = x_{B'}$ in the example above). By following the path from that node to the node x_A in $\mathcal{G}^\Sigma_{\mathcal{T}_2}$ and by constructing conjunctions over all the failing possibilities to fulfill the simulation conditions ($B_1 \sqcap B_2$ in the example above) one can construct an example inclusion $C \sqsubseteq A$ (or $A \sqsubseteq C$) that matches the difference witness A. The correctness of the algorithm described above can be seen by using Lemma 2. It is known that such concepts C can be of exponential size [5], and consequently, we cannot hope to devise an algorithm that is guaranteed to run in polynomial time.

4. COMPARISON OF APPROACHES

We now compare the hypergraph-based approach with the previous method for detecting logical differences that is developed in [5]. The previous approach also makes use of the fact that it is sufficient to search for left- and right-hand side witnesses to decide whether a logical difference exists. For computing left-hand side witnesses, the method described in [5] is similar to checking for the existence of a forward simulation. The two simulation notions are virtually identical with the difference that we work with hypergraphs, whereas canonical models are used in [5].

Fundamental differences can be found regarding the computation of right-hand side witnesses. Recall from Section 2.3 that $A \in \mathsf{cWtn}^{\mathsf{rhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ iff there exists a Σ-concept C such that $\mathcal{T}_1 \models C \sqsubseteq A$ but $\mathcal{T}_2 \not\models C \sqsubseteq A$. The general aim of [5] is to find a complete representation of all Σ-concepts C with $\mathcal{T}_2 \not\models C \sqsubseteq A$. Note that typically infinitely many such concepts C exist. For every $n \geq 0$, *finite* sets $\mathsf{noimply}^n_{\mathcal{T}_2,\Sigma}(A)$ of $\mathcal{EL}_\Sigma$-concepts are inductively defined which have the property that there exists an $\mathcal{EL}_\Sigma$-concept C with $\mathcal{T}_1 \models C \sqsubseteq A$ and $\mathcal{T}_2 \not\models C \sqsubseteq A$ iff there exists $n \geq 0$ and a $D \in \mathsf{noimply}^n_{\mathcal{T}_2,\Sigma}(A)$ such that $\mathcal{T}_1 \models D \sqsubseteq A$. The parameter n represents the maximal number of nestings of existential restrictions in C.

Two different algorithms are then presented in [5] for handling the depth parameter n. Algorithm 1 makes use of reasoning on ABoxes, i.e. finite sets of *assertions* of the form $A(c)$ or $r(c_1, c_2)$, where A is a concept name, r a role name, and c, c_1, c_2 are constants. For a TBox $\mathcal{T}$, an ABox $\mathcal{A}$ and a constant c we write $(\mathcal{T}, \mathcal{A}) \models A(c)$ iff every model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$ fulfills $c^\mathcal{I} \in A^\mathcal{I}$. The infinite sequence $\mathsf{noimply}^n_{\mathcal{T}_2,\Sigma}(A)$, $n \geq 0$, is now encoded into a polynomial-size ABox $\mathcal{A}_{\mathcal{T}_2,\Sigma}$. In this way a reduction of the original problem to an instance checking problem for the knowledge base $(\mathcal{T}_1, \mathcal{A}_{\mathcal{T}_2,\Sigma})$ can be obtained. It can be shown that $A \in \mathsf{cWtn}^{\mathsf{rhs}}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ iff $(\mathcal{T}_1, \mathcal{A}_{\mathcal{T}_2,\Sigma}) \models A(\xi)$ for some con-

stant ξ which occurs in $\mathcal{A}_{\mathcal{T}_2,\Sigma}$ and which is connected to A (in some specific sense). The ABox $\mathcal{A}_{\mathcal{T}_2,\Sigma}$ can be seen as an encoding of the infinite sequence $\mathsf{noimply}^n_{\mathcal{T}_2,\Sigma}(A)$ for $n \geq 0$; Algorithm 1 also works for cyclic terminologies, but one of its drawbacks is that for typical terminologies and large Σ, the ABox $\mathcal{A}_{\mathcal{T}_2,\Sigma}$ is of quadratic size in $\mathcal{T}_2$, which makes it more challenging to obtain an implementation that can compare *very large* terminologies together with large signatures Σ. Also, it is not straightforward to extract examples of $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ which correspond to right-hand side witnesses from an instance checking algorithm.

Algorithm 2 uses a dynamic programming approach to derive conditions that allow us to identify which concepts in $\mathsf{noimply}^n_{\mathcal{T}_2,\Sigma}(A)$ are relevant for deciding whether A is a right-hand side witness. This approach has been implemented in the logical difference tool CEX [6], which can compare large terminologies like SNOMED CT on large signatures Σ in reasonable time (cf. [5] for further details). Additionally, it is possible to extend Algorithm 2 in such a way that it becomes possible to construct examples of differences that correspond to right-hand side witnesses (which is also implemented in version 2.5 of CEX). As drawbacks, however, we have to note that this approach only works for acyclic terminologies and that possible extensions to more expressive description logics are rather challenging as the complexity and the number of the conditions that have to be checked to find right-hand side witnesses for $\mathcal{EL}$ extended with role inclusions and domain/range restrictions is already rather involved.

On the other hand, the approach presented in this paper works for cyclic TBoxes, and it benefits from the fact that the same technique, i.e. checking for the existence of certain simulations, can be used both for finding left- and right-hand side witnesses. The structures that are simulated immediately correspond to the TBoxes involved (hyperedges correspond to axioms). Moreover, the conditions that have to be fulfilled for a node to simulate another node are fairly straightforward in the sense that they only depend either on the structure of the graph, or on the logical entailment of Σ-concept names. Note that such conditions on the entailment of concept names are also present in Algorithm 1 and 2. However, the practical usefulness of our approach will still have to be demonstrated in an experimental evaluation.

5. CONCLUSION

We have presented a novel approach to the logical difference problem using a hypergraph representation of ontologies. As ontologies we consider (possibly cyclic) terminologies given in the description logic $\mathcal{EL}$. As differences between terminologies we only consider $\mathcal{EL}$-concept inclusions formulated in a given signature. A terminology is transformed into a hypergraph by taking the signature symbols as nodes and treating the axioms as hyperedges. We have devised two simulation notions between hypergraphs. The existence of the simulations is equivalent to the fact that every concept inclusion which is formulated in the considered signature and which follows from the first corresponding terminology also follows from the second terminology. Checking for the existence of simulations is tractable, confirming the established complexity bounds in [7]. If a simulation does not exist, we have sketched how to construct a concept inclusion witnessing a difference using the hypergraph. We have also discussed how the hypergraph-based approach simplifies previous approaches to computing the logical difference that required a combination of different methods.

In this paper we have considered $\mathcal{EL}$-terminologies only. This serves to illustrate the approach to the logical difference problem based on hypergraphs, but extensions to richer logics are possible. For instance, dealing with the bottom concept, role inclusions and domain and range restrictions of roles should not pose any problem. An extension to general $\mathcal{EL}$-TBoxes and even to Horn $\mathcal{SHIQ}$ ontologies would be interesting. It remains to be seen whether and in how far the form and the number of concepts witnessing a logical difference can be restricted, analogous to the primitive witness theorem (cf. Theorem 1). In any case the hypergraph and the simulation notion would need to be adapted to the richer logic, but checking for the existence of a simulation may not be tractable anymore. We leave this for future work as well as a performance evaluation of the current approach and any of its extensions on real-life ontologies. We also envision to integrate our approach for detecting logical differences into the OWL-API and into popular ontology editors such as Protégé.

6. REFERENCES

[1] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. of IJCAI-05*. Morgan-Kaufmann Publishers, 2005.

[2] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The description logic handbook: theory, implementation, and applications*. Cambridge University Press, 2007.

[3] S. Brandt. Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and—what else? In *Proc. of ECAI-04*, pages 298–302. IOS Press, 2004.

[4] E. Clarke and H. Schlingloff. Model checking. In *Handbook of Automated Reasoning*, volume II, chapter 24, pages 1635–1790. Elsevier, 2001.

[5] B. Konev, M. Ludwig, D. Walther, and F. Wolter. The logical difference for the lightweight description logic $\mathcal{EL}$. *JAIR*, 44:633–708, 2012.

[6] B. Konev, M. Ludwig, and F. Wolter. Logical difference computation with CEX2.5. In *Proc. of IJCAR-12*, pages 371–377. Springer, 2012.

[7] B. Konev, D. Walther, and F. Wolter. The logical difference problem for description logic terminologies. In *Proc. of IJCAR-08*, pages 259–274. Springer, 2008.

[8] D. Lembo, V. Santarelli, and D. F. Savo. Graph-based ontology classification in OWL 2 QL. In *Proc. of ESWC 2013*, volume 7882 of *LNCS*, pages 320–334. Springer, 2013.

[9] C. Lutz and F. Wolter. Deciding inseparability and conservative extensions in the description logic $\mathcal{EL}$. *JoSC*, 45(2):194–228, Feb. 2010.

[10] R. Nortje, A. Britz, and T. Meyer. Module-theoretic properties of reachability modules for SRIQ. In *Proc. of DL-13*, pages 868–884. CEUR-WS.org, 2013.

[11] B. Suntisrivaraporn. *Polynomial time reasoning support for design and maintenance of large-scale biomedical ontologies*. PhD thesis, TU Dresden, Germany, 2009.