Masterarbeit

Concept-By-Example in $\mathcal{EL}$ Knowledge Bases

Maurice Funk
mfunk@uni-bremen.de

18. Juli 2019

Erstgutachter: Dr. Jean C. Jung
Zweitgutachter: Prof. Dr. Sebastian Maneth

Abstract

In dieser Arbeit wird das Concept-by-Example-Problem in Wissensbasen, die in der Beschreibungslogik $\mathcal{EL}$ formuliert sind, untersucht. Bei diesem Problem gibt der Benutzer positive und negative Individuen als Beispiele an. Daraufhin soll ein $\mathcal{EL}$ Konzept gelernt werden, welches die positiven Beispiele beschreibt, aber nicht die negativen. Automatisches Lernen von Konzepten unterstützt Benutzer beim Entwickeln von Wissensbasen sowie beim Schreiben von Konzeptanfragen und wird bereits mit verschiedenen Ansätzen umgesetzt. Wir zeigen erstmals, dass zu entscheiden ob ein solches Konzept existiert, ExpTime-vollständig ist und dass ein Algorithmus, der ein solches Konzept lernt doppelt exponentielle Laufzeit haben muss. Wenn man das zu lernende Konzept auf eine bestimmte Rollentiefe beschränkt, zeigen wir NP-Vollständigkeit des Entscheidungsproblems und geben einen Exponentialzeit-Algorithmus, der das Konzept lernt, an.

In this thesis, we look at the concept-by-example problem in $\mathcal{EL}$ knowledge bases. The problem consists of automatically learning an $\mathcal{EL}$ concept that describes user provided positive examples, but does not describe user provided negative examples with regard to a given $\mathcal{EL}$ knowledge base. Concept learning supports the user during knowledge base engineering and concept query writing, and is actively pursued with multiple approaches. We show that deciding, if such a concept exists, is ExpTime-complete and that an algorithm that learns such a concept must take double exponential time in the worst case. If one restricts the concept that is learned to a fixed role depth, we show that the decision problem is only NP-complete and give an exponential time algorithm to learn it.

Universität Bremen

Nachname **Funk** Matrikelnr. **4001060**

Vorname/n **Maurice**

Diese Erklärungen sind in jedes Exemplar der Bachelor- bzw. Masterarbeit mit einzubinden.

Urheberrechtliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.
Alle Stellen, die ich wörtlich oder sinngemäß aus anderen Werken entnommen habe, habe ich unter Angabe der Quellen als solche kenntlich gemacht.

Datum Unterschrift

Erklärung zur Veröffentlichung von Abschlussarbeiten

Die Abschlussarbeit wird zwei Jahre nach Studienabschluss dem Archiv der Universität Bremen zur dauerhaften Archivierung angeboten.
Archiviert werden:

1) Masterarbeiten mit lokalem oder regionalem Bezug sowie pro Studienfach und Studienjahr 10 % aller Abschlussarbeiten
2) Bachelorarbeiten des jeweils der ersten und letzten Bachelorabschlusses pro Studienfach und Jahr.

- (●) Ich bin damit einverstanden, dass meine Abschlussarbeit im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.
- ( ) Ich bin damit einverstanden, dass meine Abschlussarbeit nach frühestens 30 Jahren (gem. §7 Abs. 2 BremArchivG) im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.
- ( ) Ich bin nicht damit einverstanden, dass meine Abschlussarbeit im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.

Datum Unterschrift

I've heard it said that a computer scientist is a mathematician who only knows how to prove things by induction.

— Steven S. Skiena, The Algorithm Design Manual

Contents

List of Figures

1 Introduction

The importance of machine processable knowledge representation rises together with the spread of computers and the internet into more aspects of human life. In recent years, a prominent knowledge representation language has been the Web Ontology Language (OWL), which was first recommended as a standard for modeling knowledge bases by the World Wide Web Consortium in 2004.[1] OWL has since been used to create knowledge bases for applications in many fields like biology, medicine, software engineering, knowledge management and cognitive systems [35, 38, 21]. OWL is employed in thousands of knowledge bases that contain billions of facts [14].

From a theoretical point of view, the basis of OWL are *description logics* [5]. Description logics are a family of logics that are decidable fragments of first-order logic, commonly written in a variable-free syntax. They are closely related to modal logic. One such description logic is called $\mathcal{EL}$. Description logics talk about the relationships of *concept descriptions* that represent important notions of an application domain. A description logic knowledge base could contain the following *terminological* knowledge about concepts ("Student", "Human", "Professor", "Lecture") and their relationships in the form of concept inclusions:

$$\text{Student} \sqsubseteq \text{Human} \sqcap \exists \text{attends.Lecture}$$
$$\text{Professor} \sqsubseteq \text{Human} \sqcap \exists \text{gives.Lecture}$$

This expresses that a student is a human who attends at least one lecture and that a professor is a human who gives at least one lecture. Here "Student", "Human", "Professor", and "Lecture" are simple concept names. These can be combined into more complicated concepts like "Student $\sqcap$ Professor" (all things that are both students and professors). This terminological part of a knowledge base is called TBox. The formal semantics of concepts of the description logic $\mathcal{EL}$ are introduced later in Section 3.1.

Furthermore, a description logic knowledge base contains *assertional* knowledge about specific objects or *individuals*:

$$\text{Student(alice)}, \quad \text{Student(bob)}, \quad \text{Lecture(logic)},$$
$$\text{attends(bob, logic)}, \quad \exists \text{attends.Seminar(alice)}$$

This expresses that the individuals Alice and Bob are students, Logic is a lecture, Bob attends the lecture Logic and Alice attends something that is described by the concept name Seminar. This assertional part of a knowledge base is called ABox.

The different description logics vary in their *expressiveness*, that is what kind of knowledge they can express. In this thesis, we consider knowledge bases and concepts

[1] `http://www.w3.org/OWL`

written in the inexpressive description logic $\mathcal{EL}$, which forms the basis of the OWL 2 EL profile. $\mathcal{EL}$ was first introduced by Baader, Küsters, and Molitor [6] as a description logic that only allows conjunctions ($\sqcap$) and existential restrictions ($\exists$). Later it was shown that important reasoning problems with general terminological knowledge can be decided in polynomial time in $\mathcal{EL}$ knowledge bases [13]. This differentiates $\mathcal{EL}$ from more expressive description logics, where subsumption checking and other reasoning problems lie in intractable complexity classes.

Although $\mathcal{EL}$ has restricted expressive power, there are large knowledge bases like SNOMED CT[2] or GALEN [34] that contain ten-thousands of concepts and their inclusions, which can be expressed in $\mathcal{EL}$ or slight extensions thereof. In order to automatically reason in knowledge bases of this size, efficient polynomial time reasoners like CEL [7] or ELK [24] were developed. These often only support terminological reasoning without any assertional knowledge.

Common reasoning tasks that these solvers and knowledge base enhanced systems in general perform, are:

- *subsumption checking*: does the terminological knowledge imply that every "A" is also a "B"? In our example assertional and terminological knowledge, this is true for "Student" and "Human", since every student is a human, but not for "Human" and "Professor", since there could be humans that are not professors.

- *instance retrieval*: which individuals are described by a concept? Here the concept is a *query* posed to the reasoner, similar to a query to a traditional database system. For the concept query $\exists$attends.Lecture the answer would be $\{\text{alice}, \text{bob}\}$ for our example assertional and terminological knowledge.. Bob is included since the assertional knowledge contains that he attends the "Lecture" logic. Alice is included since the assertional knowledge contains that she is a student and the terminological knowledge contains that every student attends a lecture.

In this thesis, we consider the reasoning task of *concept-by-example*, which can be considered the opposite of instance retrieval. One problem of instance retrieval is that users have difficulty formulating queries in large and complicated knowledge bases. Giving examples of what the answer to a query should contain and examples of what it should not contain, however, is easy. Thus a system might support the user by *learning* a concept from positive and negative examples provided by the user. The examples are *individual names* that are already known to the knowledge base. In the example from above, the user could select Alice and Bob as positive examples and Logic as a negative example. A concept that applies to the positive examples but not the negative examples would be "Human". Thus, the learned concept in this case could be "Human". The idea is that the learned concept generalizes the knowledge about the positive examples and can be used as a query to retrieve similar individuals in the future. Some reasoning settings are only interested in a set of positive examples, others consider both positive and negative examples. Note that it is not always required for the learned concept to exactly

[2] `http://www.snomed.org`

cover or not cover all examples, some definitions allow a certain degree of approximation, especially in settings where computing an exact concept may be infeasible [6].

This approach to engineering queries originates in the field of relational databases, there called *query-by-example* and was first introduced in a slightly different very early variant by Zloof [41]. Motivated by non-expert users and unfamiliar database schemata, query-by-example learns a database query that produces all positive examples but none of the negative ones. For relational databases, Query-by-example is a traditionally studied problem. Its idea has been extended to knowledge bases that contain terminological background knowledge in addition to relational data as concept-by-example.

Learning concepts from sets of examples has been the topic of some research effort in the past, as it allows for a different approach to knowledge base engineering. Instead of designing it *top-down*, by first declaring the abstract concepts of a domain and then classifying the relevant objects, one can go *bottom-up* and find the general concepts that describe relevant sets of objects. Thus it can be said that concept-by-example supports the user both in writing queries and in engineering knowledge bases.

There already are multiple approaches to learning concepts in knowledge bases. One of the most prominent ones is an approach based on refinement operators and inductive logic programming, which resulted in learning algorithms like DL-Learner [14] and DL-Foil [17]. Another important approach applies the least-common-subsumer and most-specific-concept operations [6]. However, although implementations of learning algorithms exist, no formal investigation of the *complexity* of concept learning in $\mathcal{EL}$ knowledge bases or the condition of existence of a separating concept has been done yet, as far as we are aware.

In this thesis, the following questions are considered for $\mathcal{EL}$ knowledge bases and $\mathcal{EL}$ concepts:

- *Separability*: Is there a concept that *separates* the positive from the negative examples? Or is it impossible to do so, given only the knowledge in the knowledge base? Is there a fast algorithm to decide this?
- *Concept Learning*: What is the concept that separates the positive from the negative examples? How large could it be? Is there a fast algorithm to learn a separating concept?

As an additional example we can consider the assertional and terminological knowledge from earlier and select Alice as a positive and Bob as a negative example. They are separated by the concept $\exists$attends.Seminar since only Alice attends a seminar, Bob does not.

1.1 Contributions

The first result of this thesis is a *model-theoretic characterization* of $\mathcal{EL}$ concept separability in $\mathcal{EL}$ knowledge bases. Based on this characterization we show an ExpTime lower and upper bound on the time complexity of deciding if there is a concept that separates the

positive and negative examples. Furthermore, we show that the separating $\mathcal{EL}$ concept can be of double exponential size. It follows that an algorithm that learns a separating concept must take at least double exponential time, since it has to output the concept in its entirety.

Considering that $\mathcal{EL}$ is an inexpressive description logic, this is a negative result since such high complexity was not expected. Motivated by the high complexity we look at restrictions of concept-by-example and argue that a lot of them do not improve the lower bound. However, restricting the role depth of the separating concept does improve it. Since concepts of lower role depth are easier to understand in the application domain, and thus we are interested in learning them, this seems like a natural restriction. Furthermore, real life $\mathcal{EL}$ knowledge bases often do not have enough role depth to lead to very deep concepts in the common case.

We define a variant of concept-by-example where the separating concept may be at most of a fixed role depth and provide a characterization of it. We show that deciding separability for this problem is only NP-complete. Furthermore, there is an exponential time algorithm for learning a separating concept given this restriction.

These results can help understand the behavior of existing concept learning algorithms and hint at the reasons for their time complexity. New algorithms could potentially be based on the characterization of separability given in this thesis.

Parts of this thesis were already published in Funk et al. [18].

1.2 Overview

The chapters of this thesis are structured as follows:

- Chapter 2 discusses previous work that is related to the goal of concept learning in general and concept learning for $\mathcal{EL}$ knowledge bases in particular.
- Chapter 3 formally introduces the description logic $\mathcal{EL}$ and its important properties, including standard constructions and theorems that are needed in the following chapters.
- In Chapter 4 we investigate the concept-by-example problem for $\mathcal{EL}$ with an unbounded concept role depth and establish matching upper and lower bounds for separability and learning.
- In Chapter 5 we examine the variant of concept-by-example with bounded role-depth.
- Chapter 6 discusses the results obtained in this thesis and poses possible future research questions and directions.

2 Related Work

In this chapter we examine work that is related to concept learning in knowledge bases. First we look at the distantly related field of *ontology learning*, where learning focuses on general concept inclusions instead of single concepts. Then we discuss learning queries in databases, where no background knowledge in the form of a TBox needs to be considered. Next comes the approach to learn concepts via techniques taken from inductive logic programming, which is implemented in multiple software tools. Finally we look at an approximating approach and consider the complexity results related to concept learning which have already been shown.

2.1 Ontology Learning

While concept learning only learns a single concept from positive and negative examples, *ontology learning* learns new general concept inclusions from data. Its goal is to (partially) automate the effort of writing the background terminological knowledge of a knowledge base. Sazonau [37] introduces and motivates this problem. They present and evaluate an algorithm, called *DL-Miner*, that learns additional TBox axioms conceptualizing the data in the ABox.

Another approach is to learn knowledge bases by querying an oracle. In applications, this oracle represents a domain expert, who answers questions posed by the learning algorithm, which thus learns the domain by constructing concept inclusions. Konev et al. [25] give an overview over this setting and show that an $\mathcal{EL}$ knowledge base cannot be learned with a polynomial amount of queries to the oracle.

2.2 Learning Queries in Databases

The problem of reverse engineering queries for relational databases has received some attention in the past. An important variant of this is *query-by-example*: Given a relational database D, a set of positive examples and a set of negative examples, it asks if there is a query q in some query language such that the evaluation of q on D contains all positive examples but none of the negative examples. Zloof [41] proposed this approach to make query engineering for non-expert users easier.

A common query language for this are *conjunctive queries* [9], however other settings like SPARQL queries over RDF data [2], path queries over graph databases [11] or full first order queries [1] have been investigated. Applications of this problem are similar to concept-by-example, it allows non-expert users to engineer queries and explore data in an unknown data set. For conjunctive queries and relational databases, the

query-by-example problem is known to be coNExpTime-complete [39]. Barceló and Romero [9] introduce several relaxations that make the problem tractable.

In many ways, query-by-example is very similar to the problem considered in this thesis, concept-by-example. The main differences lie in the choice of an $\mathcal{EL}$ knowledge base instead of a relational database and $\mathcal{EL}$ concepts as a "query language" instead of for example conjunctive queries.

2.3 Least Common Subsumer and Most Specific Concept

The description logic $\mathcal{EL}$ was first mentioned in the setting of learning concepts that describe positive examples. Baader, Küsters, and Molitor [6] investigate the problem of, given a set of positive examples, finding the most specific concept that describes all of them. This task can be split in two sub tasks:

- computing the *most specific concept* of a single individual, and
- computing the *least common subsumer* of a number of concepts.

Since this learning problem asks for a concept that describes a number of individuals, it seems similar to concept-for-example with no negative examples. However, concept-by-example does not result in the *most specific* concept, but any concept that describes the positive examples. In the absence of negative examples this could always be $\top$, the concept that is interpreted as all individuals. In fact, the least common subsumer and most specific concept with regard to an $\mathcal{EL}$ knowledge base do not always exist under usual semantics [3]. Zarrieß and Turhan [40] give a necessary and sufficient condition on the existence of the least common subsumer as well as an algorithm for computing it if it exists. This approach always yields a concept that generalizes all examples, if the most specific concept and least common subsumer exist.

Unfortunately this approach results in overly specific learned concepts in expressive description logics [8]. If a description logic allows concept disjunction ($\sqcup$), the least common subsumer of two concepts C_1 and C_2 simply becomes their disjunction $C_1 \sqcup C_2$. The least common subsumer thus still applies to all positive examples but does not generalize the properties of the positive examples in any meaningful way.

2.4 Inductive Logic Programming

In order to avoid the mentioned shortcomings of the most specific concept and least common subsumer approach, Badea and Nienhuys-Cheng [8] apply a learning technique from inductive logic programming to description logics. For an introduction to inductive logic programming, see Nienhuys-Cheng and De Wolf [33]. The idea is to use downward and upward *refinement operators* to search the space of possible concepts and find a concept that separates the positive and negative examples.

Here, a refinement operator is a function from concepts to sets of concepts. A downward refinement operator constructs a number of more specialized concepts (that

cover fewer individuals). A upward refinement operator constructs a number of more general concepts (that cover more individuals). A refinement operator is *finite* if it only produces a finite number of concepts. It is called *complete* if any more specialized (or more general) concept can be reached by repeated application of the operator to one of its results. A refinement operator is *proper* if one cannot reach its initial input by repeated application of the operator. An operator is *ideal* if it is finite, complete and proper [28].

Badea and Nienhuys-Cheng give specific refinement operators for the description logic $\mathcal{ALER}$ ($\mathcal{EL}$ extended with the bottom concept $\bot$, negation of atoms $\neg A$, and value restrictions $\forall$) and describe how to search for a separating concept. They discuss that learning in their framework is at least NP-hard [8].

Inspired by this approach is *DL-Learner*, a framework for learning problems in description logics. It implements several algorithms that support many features of OWL [14, 26]. For $\mathcal{EL}$ concept learning from positive and negative examples, DL-Learner implements an algorithm called EL Tree Learner that is optimized for the properties of $\mathcal{EL}$. It is based on an ideal downward refinement operator, which was shown to not exist for more expressive description logics than $\mathcal{EL}$ [28, 29]. The algorithm searches the space of all possible concepts by applying the operator in each step and evaluating the resulting more specific concepts. Since the refinement operator is complete, it eventually arrives at a separating concept, if one exists. In practice however, the algorithm is limited by a maximum run time and might not arrive at a separating concept, even if one exists [27]. No formal analysis of the time complexity of the EL Tree Learner algorithm is given.

Another similar approach that is closer to the classical learning in *inductive logic programming* is *DL-Foil*, which adapts the classical Foil algorithm for description logic. The algorithm computes generalizations for subsets of the positive examples that do not cover any negative examples via a downward refinement operator and joins them together via a disjunction to cover all positive examples [17]. Since $\mathcal{EL}$ does not allow disjunctions of concepts, the DL-Foil approach cannot be applied to the setting of this thesis.

2.5 Other Approaches

Sarker and Hitzler [36] present a different approach to concept learning that aims to reduce the number of calls to an external description logic reasoner since these can lead to performance bottlenecks for expressive description logics. To achieve this, it does not use refinement operators but precomputes the extensions of a number of heuristically chosen concept expressions. It then tries to combine the precomputed concepts using only negation, conjunction and disjunction to cover all examples. This approach is approximate, as it may not find an exact cover of the examples, but is sufficiently precise for the specific type of knowledge base on which the authors aim to improve performance of concept learning. This demonstrates that fast concept learning is of interest and that sacrificing completeness to achieve it may be a possibility.

Jung et al. [22] investigate concept learning in first-order structures for the description logic horn-$\mathcal{ALC}$, which is a restriction of $\mathcal{ALC}$ to a fragment of Horn first order logic.

Since first-order structures can be seen as ABoxes, their results apply to concept learning in knowledge bases without a TBox. They give a characterization of concept separability based on *Horn simulations* and show that deciding separability for horn-$\mathcal{ALC}$ is ExpTime-complete. This does not change if one restricts the depth of the separating horn-$\mathcal{ALC}$ concept to l as part of the input. The variant of this problem that restricts the depth of the separating horn-$\mathcal{ALC}$ concept to l is still ExpTime-complete if l is given in binary encoding as part of the input.

Gutiérrez-Basulto, Jung, and Sabellek [19] extend the approach of ten Cate and Dalmau [39] from relational databases to ontology-enriched systems, that is knowledge bases with a TBox, and consider (unions of) conjunctive queries as a query language. For the description logic Horn-$\mathcal{ALC}$ as a knowledge base language they show that the problem of deciding if a separating conjunctive query exists is coNExpTime-complete.

Beyond the results concerning $\mathcal{EL}$ in this thesis, Funk et al. [18] study concept separability for more expressive description logics, in combinations of knowledge base language and concept language.

The relevant results can be summarized as

- $\mathcal{EL}$-concept separability is ExpTime-complete in $\mathcal{EL}$ and $\mathcal{ELI}$ knowledge bases,
- $\mathcal{ELI}$-concept separability is undecidable in $\mathcal{EL}$ and more expressive knowledge bases,
- $\mathcal{ALC}$-concept separability is NExpTime-complete,
- $\mathcal{ALCQI}$-concept separability is ExpTime-complete.

Note that this is, as far as we are aware, the first publication that investigates the complexity of deciding concept separability in description logic knowledge bases with general TBoxes.

3 Preliminaries

In this chapter we give definitions, constructions, and important theorems needed for the following chapters. The following topics are covered:

- the description logic $\mathcal{EL}$,
- simulation relations between interpretations,
- canonical models for $\mathcal{EL}$ knowledge bases,
- and constructing interpretations from concepts and vice versa.

3.1 The Description Logic $\mathcal{EL}$

Concepts Description logics describe knowledge of a domain with *concepts*. $\mathcal{EL}$ concepts are defined using a set of concept names N_C and a set of role names N_R. Generally throughout this thesis, we will use A and B as some concept names from N_C, while we use r for role names from N_R. All concept names $A \in N_C$ and $\top$ are $\mathcal{EL}$ concepts. If C and D are $\mathcal{EL}$ concepts, then

- $C \sqcap D$ (conjunction) and
- $\exists r.C$ for all role names $r \in N_R$ (existential restriction)

are also $\mathcal{EL}$ concepts. We will use C and D if we want to refer to these *complex* concepts.

The semantics of an $\mathcal{EL}$ concept is defined in terms of interpretations $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$. The domain $\Delta^\mathcal{I}$ of an interpretation $\mathcal{I}$ is a non empty set of individuals and the interpretation function $\cdot^\mathcal{I}$ maps concept names $A \in N_C$ to sets of individuals $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ and role names $r \in N_R$ to relations $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. It can be inductively extended to complex $\mathcal{EL}$ concepts as follows:

$$\begin{aligned} \top^\mathcal{I} &= \Delta^\mathcal{I} \\ (C \sqcap D)^\mathcal{I} &= C^\mathcal{I} \cap D^\mathcal{I} \\ (\exists r.C)^\mathcal{I} &= \{d \in \Delta^\mathcal{I} \mid (d, e) \in r^\mathcal{I} \wedge e \in C^\mathcal{I}\} \end{aligned}$$

We call $C^\mathcal{I}$ the *extension* of C.

The depth of the nesting of existential restrictions in an $\mathcal{EL}$ concept C is called its *role depth*. We define the role depth $\mathrm{depth}(C)$ inductively by

$$\begin{aligned} \mathrm{depth}(A) = \mathrm{depth}(\top) &= 0 \\ \mathrm{depth}(C \sqcap D) &= \max(\mathrm{depth}(C), \mathrm{depth}(D)) \\ \mathrm{depth}(\exists r.C) &= \mathrm{depth}(C) + 1. \end{aligned}$$

Another important property of an $\mathcal{EL}$ concept C is its *size* $|C|$. It roughly corresponds to the number of symbols that are needed to write a concept down and can serve as an upper bound for, for example, the number of conjunctions in a concept. We can define it inductively in a similar way:

$$|A| = |\top| = 1$$
$$|C \sqcap D| = |C| + |D| + 1$$
$$|\exists r.C| = |C| + 3$$

In a given interpretation $\mathcal{I}$, we call the individual $e \in \Delta^{\mathcal{I}}$ an r-successor of another individual $d \in \Delta^{\mathcal{I}}$ if $(d, e) \in r^{\mathcal{I}}$ for some role name $r \in N_R$. The r-predecessors of an individual are defined analogously. The outdegree $\deg(e)$ of an individual $e \in \Delta^{\mathcal{I}}$ is the total number of its successors:

$$\deg(e) = |\{e' \in \Delta^{\mathcal{I}} \mid (e, e') \in r^{\mathcal{I}} \text{ for all } r \in N_r\}|.$$

The maximum outdegree $\deg(\mathcal{I})$ of an interpretation $\mathcal{I}$ is the maximum of the outdegrees of its individuals:

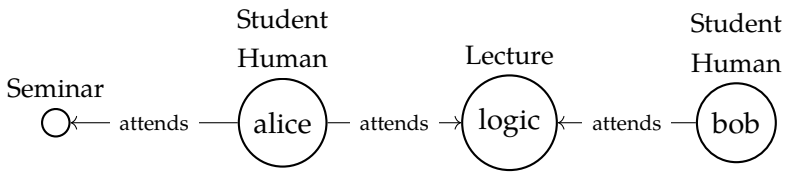
$$\deg(\mathcal{I}) = \max\{\deg(e) \mid e \in \Delta^{\mathcal{I}}\}.$$

Knowledge bases An *$\mathcal{EL}$ knowledge base* $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ consists of an $\mathcal{EL}$ TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. A *TBox* is a finite set of concept inclusions $C \sqsubseteq D$ that represent *terminological* knowledge about a domain by stating the relationship of the $\mathcal{EL}$ concepts C and D. An interpretation $\mathcal{I}$ satisfies the concept inclusion $C \sqsubseteq D$, written $\mathcal{I} \models C \sqsubseteq D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. An *ABox* is a finite set of concept assertions and role assertions that are defined over a set of *individual names* N_I. *Concept assertions* are of the form $C(a)$, where C is an $\mathcal{EL}$ concept and $a \in N_I$ is an individual name. In order to interpret concept assertions, we extend the interpretation function $\cdot^{\mathcal{I}}$ of an interpretation $\mathcal{I}$ to map individual names a to individuals $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. An interpretation $\mathcal{I}$ then satisfies a concept assertion $C(a)$, written $\mathcal{I} \models C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$. *Role assertions* are of the form $r(a, b)$, where r is a role name and $a, b \in N_I$ are individual names. An interpretation $\mathcal{I}$ satisfies the role assertion $r(a, b)$, written $\mathcal{I} \models r(a, b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$. We use $\text{ind}(\mathcal{A}) \subset N_I$ to denote the finite set of all individual names that are used in an ABox $\mathcal{A}$. Note that in some situations in this thesis, the mapping from individual names to individuals is injective and the explicit mapping of the name a to the individual $a^{\mathcal{I}}$ is left out, if $\mathcal{I}$ is clear from context.

An interpretation that satisfies all concept inclusions of a TBox $\mathcal{T}$ is called a *model* of $\mathcal{T}$. Similarly, an interpretation that satisfies all assertions of an ABox $\mathcal{A}$ is called a *model* of $\mathcal{A}$. If $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$, it is also called a model of the knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. If a concept inclusion $C \sqsubseteq D$ is satisfied in all models of a TBox $\mathcal{T}$ (or a knowledge base $\mathcal{K}$), we write $\mathcal{T} \models C \sqsubseteq D$ (or $\mathcal{K} \models C \sqsubseteq D$). Similarly, if a concept assertion $C(a)$ is satisfied in all models of a knowledge base $\mathcal{K}$, we write $\mathcal{K} \models C(a)$.

Again, it can be useful to talk about the *size* $|\mathcal{K}|$ of a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ since it can serve as an upper bound for other properties. It is simply the sum of the sizes of its components, the ABox and TBox:

$$|\mathcal{K}| = |\mathcal{T}| + |\mathcal{A}|.$$



$$\mathcal{T} = \{\text{Student} \sqsubseteq \text{Human} \sqcap \exists \text{attends}.\text{Lecture}\}$$
$$\mathcal{A} = \{\text{Student}(\text{alice}), \quad \text{Student}(\text{bob}), \quad \text{Lecture}(\text{logic}),$$
$$\text{attends}(\text{bob}, \text{logic}), \quad \exists \text{attends}.\text{Seminar}(\text{alice})\}$$

Figure 3.1: A model of the example knowledge base

The size of an ABox $|\mathcal{A}|$ is the number of role assertions and the sum of the size of its concept assertions:

$$|\mathcal{A}| = \sum_{r(a,b) \in \mathcal{A}} 1 + \sum_{C(a) \in \mathcal{A}} (1 + |C|).$$

The size of a TBox $|\mathcal{T}|$ is the size of all the concept inclusions it contains:

$$|\mathcal{T}| = \sum_{C \sqsubseteq D \in \mathcal{T}} |C| + |D| + 1.$$

Figure 3.1 shows an example knowledge base similar to the one from the introduction and an interpretation that is a model of that knowledge base. The example TBox contains one concept inclusions and the ABox consists of four concept assertions and one role assertion. The interpretation is represented as a directed graph, where the individuals are nodes, roles are labeled edges and concept names are written next to a node if its extension contains the corresponding individual. Note how Alice and Bob fulfill the concept inclusion by being Human and attending the Lecture logic.

A *signature* Σ is a finite set of concept names and role names $\Sigma \subseteq N_C \cup N_R$. The signature $\text{sig}(\mathcal{K})$ of a knowledge base $\mathcal{K}$ is the set of concept and role names that are used in $\mathcal{K}$. The interpretation $\mathcal{I}'$ is the Σ-restriction of an interpretation $\mathcal{I}$, if

$$A^{\mathcal{I}'} = A^{\mathcal{I}} \text{ for all } A \in \Sigma$$
$$r^{\mathcal{I}'} = r^{\mathcal{I}} \text{ for all } r \in \Sigma$$
$$A^{\mathcal{I}'} = \emptyset \text{ for all } A < \Sigma$$
$$r^{\mathcal{I}'} = \emptyset \text{ for all } r < \Sigma.$$

Normal form Some proofs and constructions regarding an $\mathcal{EL}$ knowledge base $\mathcal{K}$ become simpler if one considers it to be in normal form:

Definition 3.1. *An $\mathcal{EL}$ knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is in normal form if all concept inclusions in $\mathcal{T}$ have one of the following forms*

$$\begin{aligned} A_1 \sqcap A_2 &\sqsubseteq B \\ A_1 &\sqsubseteq \exists r.B \\ \exists r.A_1 &\sqsubseteq B \end{aligned}$$

where $A_1, A_2, B \in N_C \cup \{\top\}$, and all concept assertions in $\mathcal{A}$ are of the form

$$A(a)$$

where $A \in N_C$.

Baader, Brandt, and Lutz [4] give six normalization rules that transform am $\mathcal{EL}$ TBox into its normal form $\widehat{\mathcal{T}}$ by introducing new concept names from N_C that do not occur in $\text{sig}(\mathcal{K})$. Furthermore a concept assertion $C(a)$ from $\mathcal{A}$ can be transformed into normal form by introducing a new concept name $A \in N_C$ that does not occur in $\text{sig}(\mathcal{K})$ for the assertion $A(a)$ and adding the concept inclusion $A \sqsubseteq C$ to the TBox.

Since every knowledge base can be transformed into normal form, this requirement does not limit the expressiveness of $\mathcal{EL}$. The normalized knowledge base $\widehat{\mathcal{K}}$ is a *conservative extension* of $\mathcal{K}$, meaning that every model of $\widehat{\mathcal{K}}$ is also a model of $\mathcal{K}$, and every model of $\mathcal{K}$ can be extended to a model of $\widehat{\mathcal{K}}$ by adding extensions of the newly introduced concept names [4].

Reasoning The most important reasoning problem for $\mathcal{EL}$ knowledge bases, that is examined in previous literature is concept subsumption:

Input: An $\mathcal{EL}$ TBox $\mathcal{T}$, concepts C and D
Question: Does $\mathcal{T} \models C \sqsubseteq D$?

While this problem is intractable for more expressive description logics, $\mathcal{EL}$ allows for efficient subsumption checking:

Theorem 3.2. *(Brandt [13]) Let $\mathcal{T}$ be an $\mathcal{EL}$ TBox, C and D be $\mathcal{EL}$ concepts. Then subsumption $\mathcal{T} \models C \sqsubseteq D$ can be decided in polynomial time.*

Another relevant reasoning problem is *instance checking*, which checks whether a given knowledge base $\mathcal{K}$ implies that an individual is an instance of a concept:

Input: An $\mathcal{EL}$ knowledge base $\mathcal{K}$, concept C and individual name a
Question: Does $\mathcal{K} \models C(a)$?

Instance checking can also be decided in polynomial time for $\mathcal{EL}$ knowledge bases [12].

3.2 Simulations and Interpretations

We use *simulation* relations between interpretations to show properties of the concept-by-example problem in later chapters:

Definition 3.3. *A relation $\sigma \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{J}}$ between two interpretations $\mathcal{I}$ and $\mathcal{J}$ is a Σ-simulation for a signature Σ if for all $(c, d) \in \sigma$:*

- *[Atom]* $c \in A^{\mathcal{I}} \implies d \in A^{\mathcal{J}}$ *for all* $A \in \Sigma$
- *[Forth]* $(c, c') \in r^{\mathcal{I}} \implies \exists d' \in \Delta^{\mathcal{J}}.(d, d') \in r^{\mathcal{J}}$ *and* $(c', d') \in \sigma$ *for all* $r \in \Sigma$.

Some properties of simulations are:

- The union $\sigma = \sigma_1 \cup \sigma_2$ of two simulations σ_1 and σ_2 is also a simulation.
- The concatenation

$$\sigma = \sigma_1 \cdot \sigma_2 = \{(c, e) \mid (c, d) \in \sigma_1 \wedge (d, e) \in \sigma_2\}$$

  of two simulations σ_1 and σ_2 is also a simulation.

We write $\mathcal{I} \precsim_\Sigma \mathcal{J}$ if there exists a Σ-simulation between the interpretations $\mathcal{I}$ and $\mathcal{J}$. Additionally we write $(\mathcal{I}, a) \precsim_\Sigma (\mathcal{J}, b)$ for $a \in \Delta^{\mathcal{I}}$ and $b \in \Delta^{\mathcal{J}}$ if there exists a Σ-simulation σ between $\mathcal{I}$ and $\mathcal{J}$ and $(a, b) \in \sigma$. If Σ includes all relevant names, we omit it and just write $\mathcal{I} \precsim \mathcal{J}$. Since the concatenation of simulations is also a simulation, we know that $\precsim_\Sigma$ is transitive.

Simulations are a useful tool for showing properties of $\mathcal{EL}$, since they "capture" the expressiveness of $\mathcal{EL}$ concepts, as expressed in the following theorem:

Theorem 3.4. *(Lutz and Wolter [32]) Let $\mathcal{I}$ and $\mathcal{J}$ be interpretations and C be an $\mathcal{EL}$ concept. If $(\mathcal{I}, c) \precsim (\mathcal{J}, d)$, then $c \in C^{\mathcal{I}} \implies d \in C^{\mathcal{J}}$. Conversely, if $\mathcal{I}$ and $\mathcal{J}$ are finite and $c \in C^{\mathcal{I}} \implies d \in C^{\mathcal{J}}$, then $(\mathcal{I}, c) \precsim (\mathcal{J}, d)$.*

Lutz and Wolter refer to Clarke and Schlingloff [16] for a proof of this theorem in the setting of model checking.

Clarke and Schlingloff [16] give an algorithm to compute a Σ-simulation between two finite interpretations $\mathcal{I}$ and $\mathcal{J}$. It computes a sequence of relations $\sigma_0, \sigma_1, \ldots \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{J}}$ as follows:

- [Init] $(c, d) \in \sigma_0$ if and only if for all $A \in \Sigma$ it holds that $c \in A^{\mathcal{I}} \iff d \in A^{\mathcal{J}}$.
- [Step] $(c, d) \in \sigma_{n+1}$ if and only if $(c, d) \in \sigma_n$ and for all $r \in N_R$ and $c' \in \Delta^{\mathcal{I}}$ such that $(c, c') \in r^{\mathcal{I}}$, there is a $d' \in \Delta^{\mathcal{J}}$ such that $(d, d') \in r^{\mathcal{J}}$ and $(c', d') \in \sigma_n$.

If $\sigma_n = \sigma_{n+1}$ the construction terminates and $\sigma = \sigma_n$ is the largest Σ-simulation between $\mathcal{I}$ and $\mathcal{J}$. That is $(\mathcal{I}, a) \precsim_\Sigma (\mathcal{J}, b)$ if and only if $(a, b) \in \sigma$. The initial relation σ_0 contains at most $|\Delta^\mathcal{I} \times \Delta^\mathcal{J}|$ many elements and each step from σ_n to σ_{n+1} must remove at least one element from it, otherwise the algorithm would terminate. This means that the algorithm runs at most for $|\Delta^\mathcal{I} \times \Delta^\mathcal{J}|$ steps. Each step can be performed in polynomial time in the size of $\mathcal{I}$ and $\mathcal{J}$. Thus the algorithm terminates in polynomial time.

Theorem 3.5. *(Clarke and Schlingloff [16]) Let $\mathcal{I}$ and $\mathcal{J}$ be finite interpretations. A Σ-simulation between $\mathcal{I}$ and $\mathcal{J}$ can be computed in polynomial time in $|\mathcal{I}| + |\mathcal{J}|$.*

Furthermore we need some properties and operations of interpretations for the following chapters. First, we say that an interpretation $\mathcal{I}$ is *tree-shaped*, if every individual $a \in \Delta^\mathcal{I}$ has at most a single predecessor over all $r \in N_R$:

$$|\{a' \mid (a', a) \in r^\mathcal{I} \wedge r \in N_R\}| = 1$$

For each interpretation $\mathcal{I}$, there is an equivalent tree-shaped interpretation $\widehat{\mathcal{I}}$:

Let $\mathcal{I}$ be an interpretation and $d \in \Delta^\mathcal{I}$. A d-path in $\mathcal{I}$ is a finite sequence $d_0, d_1, \ldots d_{n-1}$ of $n \geq 1$ elements such that

- $d_0 = d$,
- for all $1 \leq i < n$ there is a role $r_i \in N_R$ such that $(d_{i-1}, d_i) \in r_i^\mathcal{I}$.

The *end node* of a d-path $p = d_0, d_1, \ldots, d_{n-1}$ is $\mathsf{end}(p) = d_{n-1}$.

Definition 3.6. *The unravelling of an interpretation $\mathcal{I}$ is the following interpretation $\widehat{\mathcal{I}}$:*

$$\Delta^{\widehat{\mathcal{I}}} = \{p \mid p \text{ is a } d\text{-path in } \mathcal{I} \text{ for any } d \in \Delta^\mathcal{I}\},$$
$$A^{\widehat{\mathcal{I}}} = \{p \in \Delta^{\widehat{\mathcal{I}}} \mid \mathsf{end}(p) \in A^\mathcal{I}\} \text{ for all } A \in N_C,$$
$$r^{\widehat{\mathcal{I}}} = \{(p, p') \in \Delta^{\widehat{\mathcal{I}}} \times \Delta^{\widehat{\mathcal{I}}} \mid p' = (p, \mathsf{end}(p')) \wedge (\mathsf{end}(p), \mathsf{end}(p')) \in r^\mathcal{I}\} \text{ for all } r \in N_R.$$

The unravelling of an interpretation is equivalent to the original interpretation in the sense that $\mathcal{EL}$ concepts cannot distinguish between them:

Lemma 3.7. *For all interpretations $\mathcal{I}$, $\mathcal{EL}$ concepts C and $a \in \Delta^\mathcal{I}$*

$$\mathcal{I} \models C(a) \text{ if and only if } \widehat{\mathcal{I}} \models C(a)$$

Proof. Follows from the fact that

$$\sigma = \{(\mathsf{end}(p), p) \mid p \in \Delta^{\widehat{\mathcal{I}}}\}$$

is a simulation in both directions [32] and Theorem 3.4. □

Another operation that is important for the following chapters is the *product* of two interpretations $\mathcal{I}$ and $\mathcal{J}$.

Definition 3.8. *Let $\mathcal{I}$ and $\mathcal{J}$ be interpretations. Their product interpretation $\mathcal{I} \times \mathcal{J}$ is*

$$\begin{aligned}
\Delta^{\mathcal{I}\times\mathcal{J}} &= \Delta^{\mathcal{I}} \times \Delta^{\mathcal{J}} \\
A^{\mathcal{I}\times\mathcal{J}} &= \{(d,e) \in \Delta^{\mathcal{I}\times\mathcal{J}} \mid d \in A^{\mathcal{I}} \wedge e \in A^{\mathcal{J}}\} \text{ for all } A \in N_C \\
r^{\mathcal{I}\times\mathcal{J}} &= \{((d,e),(d',e')) \in \Delta^{\mathcal{I}\times\mathcal{J}} \times \Delta^{\mathcal{I}\times\mathcal{J}} \mid (d,d') \in r^{\mathcal{I}} \wedge (e,e') \in r^{\mathcal{J}}\} \text{ for all } r \in N_R.
\end{aligned}$$

The product $(\mathcal{I}, a) \times (\mathcal{J}, b)$, $a \in \Delta^{\mathcal{I}}$ and $b \in \Delta^{\mathcal{J}}$ is defined as $(\mathcal{I} \times \mathcal{J}, (a,b))$. We write

$$\prod_{i=1}^{n} (\mathcal{I}_i, a_i) = (\mathcal{I}_1, a_1) \times \cdots \times (\mathcal{I}_n, a_n)$$

as a shorthand for the product of multiple interpretations $\mathcal{I}_1, \ldots, \mathcal{I}_n$. Furthermore it is important to note that the number of individuals of $\prod_{i=1}^{n}(\mathcal{I}_i, a_i)$ is exponential in n.

3.3 The Canonical Model of a Knowledge Base

A useful tool for working with an $\mathcal{EL}$ knowledge base $\mathcal{K}$ is its *canonical model* $\mathcal{G}_{\mathcal{K}}$. It is an interpretation that has the important property that it is *universal*, that is

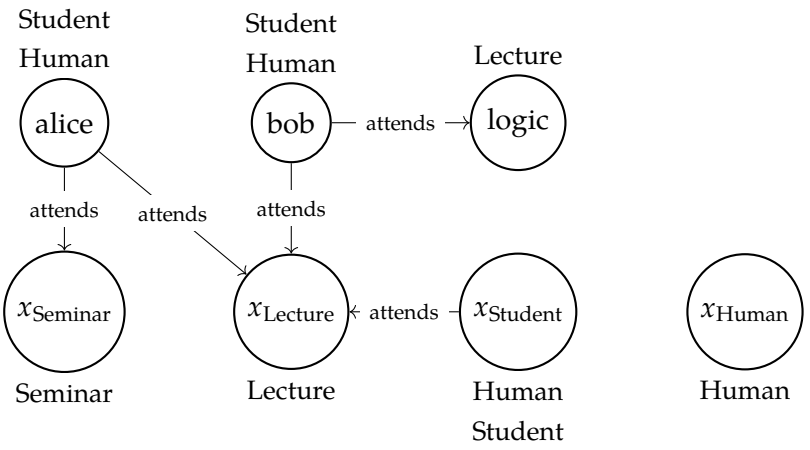
$$\mathcal{K} \models C(a) \text{ if and only if } \mathcal{G}_{\mathcal{K}} \models C(a)$$

for all $\mathcal{EL}$ concepts C and all individuals a. In the following section we will show how to construct $\mathcal{G}_{\mathcal{K}}$ from $\mathcal{K}$ and give a proof that $\mathcal{G}_{\mathcal{K}}$ indeed has this property. Lutz and Wolter [31] describe a similar canonical model for $\mathcal{EL}$ TBoxes and show its property. Lutz, Toman, and Wolter [30] give a slightly more complicated canonical model for $\mathcal{ELH}_{\bot}^{dr}$ knowledge bases. The following construction is based on the latter, modified for $\mathcal{EL}$ knowledge bases.

Definition 3.9. *The canonical model of an $\mathcal{EL}$ knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ in normal form is the interpretation $\mathcal{G}_{\mathcal{K}}$ with*

$$\begin{aligned}
N &= \{x_A \mid A \text{ is a concept name in } sig(\mathcal{K})\} \\
\Delta^{\mathcal{G}_{\mathcal{K}}} &= ind(\mathcal{A}) \cup N \\
A^{\mathcal{G}_{\mathcal{K}}} &= \{a \in ind(\mathcal{A}) \mid \mathcal{K} \models A(a)\} \cup \{x_B \in N \mid \mathcal{T} \models B \sqsubseteq A\} \\
r^{\mathcal{G}_{\mathcal{K}}} &= \{(a,b) \in ind(\mathcal{A})^2 \mid r(a,b) \in \mathcal{A}\} \\
&\quad \cup \{(a, x_A) \in ind(\mathcal{A}) \times N \mid \mathcal{K} \models \exists r.A(a)\} \\
&\quad \cup \{(x_A, x_B) \in N^2 \mid \mathcal{T} \models A \sqsubseteq \exists r.B\}
\end{aligned}$$

for all A and r in $sig(\mathcal{K})$:



$$\mathcal{T} = \{\text{Student} \sqsubseteq \text{Human} \sqcap \exists \text{attends}.\text{Lecture}\}$$
$$\mathcal{A} = \{\text{Student}(\text{alice}), \quad \text{Student}(\text{bob}), \quad \text{Lecture}(\text{logic}),$$
$$\text{attends}(\text{bob}, \text{logic}), \quad \exists \text{attends}.\text{Seminar}(\text{alice})\}$$

Figure 3.2: The canonical model $\mathcal{G}_\mathcal{K}$ of the example knowledge base

Note that the size of $\mathcal{G}_\mathcal{K}$, i.e. $|\Delta^{\mathcal{G}_\mathcal{K}}|$ is bounded by $|\mathcal{K}|$. Since concept subsumption and instance checking can be decided in polynomial time [13], the canonical model $\mathcal{G}_\mathcal{K}$ can be constructed in polynomial time in $|\mathcal{K}|$. Figure 3.2 shows the canonical model $\mathcal{G}_\mathcal{K}$ for an example knowledge base.

Since the canonical model differs in some details from the model of Lutz, Toman, and Wolter [30], we first show that it is indeed a model of $\mathcal{K}$ and then that it is universal.

Lemma 3.10. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be an $\mathcal{EL}$ knowledge base in normal form. Then $\mathcal{G}_\mathcal{K}$ is a model of $\mathcal{K}$.*

Proof. $\mathcal{G}_\mathcal{K}$ is a model of $\mathcal{A}$ since all individuals, concept assertions and role assertions are contained in $\mathcal{G}_\mathcal{K}$ by construction. It remains to be shown, that $\mathcal{G}_\mathcal{K}$ fulfills all concept inclusions in $\mathcal{T}$. We only consider concept inclusions in $\mathcal{EL}$ normal form:

Case $A_1 \sqcap A_2 \sqsubseteq B$: Consider the two possible kinds of elements in $(A_1 \sqcap A_2)^{\mathcal{G}_\mathcal{K}}$

1. Let $a \in \text{ind}(\mathcal{A}) \cap (A_1 \sqcap A_2)^{\mathcal{G}_\mathcal{K}}$. This means by construction that $\mathcal{K} \models A_1(a)$ and $\mathcal{K} \models A_2(a)$, but since $A_1 \sqcap A_2 \sqsubseteq B \in T$ it also follows that $\mathcal{K} \models B(a)$. Thus, by construction, $a \in B^{\mathcal{G}_\mathcal{K}}$.
2. Let $x_A \in N \cap (A_1 \sqcap A_2)^{\mathcal{G}_\mathcal{K}}$, by construction this means that $\mathcal{T} \models A \sqsubseteq A_1$ and $\mathcal{T} \models A \sqsubseteq A_2$ for the concept name A of x_A, but since $A_1 \sqcap A_2 \sqsubseteq B \in \mathcal{T}$, this also means that $\mathcal{T} \models A \sqsubseteq B$ and thus $x_A \in B^{\mathcal{G}_\mathcal{K}}$ by construction.

Case $B \sqsubseteq \exists r.B'$: Consider the two possible kinds of elements in $B^{\mathcal{G}_\mathcal{K}}$

1. Let $a \in \text{ind}(\mathcal{A}) \cap B^{\mathcal{G}_\mathcal{K}}$. Since $B \sqsubseteq \exists r.B' \in \mathcal{T}$ it follows that $\mathcal{K} \models \exists r.B'(a)$ holds and thus $(a, x_{B'}) \in r^{\mathcal{G}_\mathcal{K}}$. By construction of $B'^{\mathcal{G}_\mathcal{K}}$ we have that $x_{B'} \in B'^{\mathcal{G}_\mathcal{K}}$. Thus $a \in (\exists r.B')^{\mathcal{G}_\mathcal{K}}$.
2. Let $x_A \in N \cap B^{\mathcal{G}_\mathcal{K}}$. Since $x_A \in B^{\mathcal{G}_\mathcal{K}}$, we have that $\mathcal{T} \models A \sqsubseteq B$. Since $B \sqsubseteq \exists r.B' \in \mathcal{T}$ it also follows that $\mathcal{T} \models A \sqsubseteq \exists r.B'$. Thus there is $x_{B'} \in B'^{\mathcal{G}_\mathcal{K}}$ such that $(x_A, x_{B'}) \in r^{\mathcal{G}_\mathcal{K}}$. This means that $x_A \in (\exists r.B')^{\mathcal{G}_\mathcal{K}}$.

Case $\exists r.B \sqsubseteq B'$: Consider the two possible kinds of elements in $(\exists r.B)^{\mathcal{G}_\mathcal{K}}$

1. Let $a \in \text{ind}(\mathcal{A}) \cap (\exists r.B)^{\mathcal{G}_\mathcal{K}}$. This means that there exists an $b \in B^{\mathcal{G}_\mathcal{K}}$ such that $(a, b) \in r^{\mathcal{G}_\mathcal{K}}$. By construction of $\mathcal{G}_\mathcal{K}$, we thus know that $\mathcal{K} \models \exists r.B(a)$ and since $\exists r.B \sqsubseteq B' \in \mathcal{T}$, we have $\mathcal{K} \models B'(a)$ and it holds that $a \in B'^{\mathcal{G}_\mathcal{K}}$ by construction of $B'^{\mathcal{G}_\mathcal{K}}$.
2. Let $x_A \in N \cap (\exists r.B)^{\mathcal{G}_\mathcal{K}}$. This means that there is a $x_{A'} \in B^{\mathcal{G}_\mathcal{K}}$ such that $(x_A, x_{A'}) \in r^{\mathcal{G}_\mathcal{K}}$. By construction of $\mathcal{G}_\mathcal{K}$ it follows necessarily that $\mathcal{T} \models A' \sqsubseteq B$ and $\mathcal{T} \models A \sqsubseteq \exists r.A'$. Combining these two gives $\mathcal{T} \models A \sqsubseteq \exists r.B$, but since $\exists r.B \sqsubseteq B' \in \mathcal{T}$, it also follows that $\mathcal{T} \models A \sqsubseteq B'$ and thus $x_A \in B'^{\mathcal{G}_\mathcal{K}}$ by construction of $B'^{\mathcal{G}_\mathcal{K}}$. □

With this Lemma, we can prove the central property of the *canonical model* $\mathcal{G}_\mathcal{K}$:

Theorem 3.11. *For an $\mathcal{EL}$ knowledge base $\mathcal{K}$ and any $\mathcal{EL}$ concept C and any individual name $a \in ind(\mathcal{A})$*

$$\mathcal{K} \models C(a) \text{ if and only if } \mathcal{G}_\mathcal{K} \models C(a).$$

Proof. First, assume that $\mathcal{K} \models C(a)$. It directly follows that $\mathcal{G}_\mathcal{K} \models C(a)$ since $\mathcal{G}_\mathcal{K}$ is a model of $\mathcal{K}$ by Lemma 3.10. It remains to be shown that $\mathcal{K} \models C(a)$ if $\mathcal{G}_\mathcal{K} \models C(a)$. From the definition of models we have that this is equivalent to $a \in C^{\mathcal{G}_\mathcal{K}} \implies a \in C^{\mathcal{I}}$ for all $\mathcal{I}$ that are models of $\mathcal{K}$. By Theorem 3.4 we know that this is the case if $(\mathcal{G}_\mathcal{K}, a) \precsim (\mathcal{I}, a)$ for all $\mathcal{I}$.

Let's choose an arbitrary $\mathcal{I}$. In order to show that $\mathcal{K} \models C(a)$ if $\mathcal{G}_\mathcal{K} \models C(a)$, we want to prove that σ with

$$\sigma = \{(b, b) \mid b \in \text{ind}(\mathcal{A})\} \cup \{(x_A, x) \mid x \in A^{\mathcal{I}}\}.$$

is a Σ-simulation witnessing $(\mathcal{G}_\mathcal{K}, a) \precsim (\mathcal{I}, a)$. By construction of σ we have that $(a, a) \in \sigma$, it remains to be shown that σ fulfills the conditions of Σ-simulations.

For the [Atom] condition, consider a pair $(a, a) \in \sigma$ with $a \in \text{ind}(\mathcal{A})$. Assume $a \in A^{\mathcal{G}_\mathcal{K}}$ but $a <A^{\mathcal{I}}$ for contradiction for a $A \in \Sigma$. By construction of $A^{\mathcal{G}_\mathcal{K}}$ it follows that $\mathcal{K} \models A(a)$. Since $\mathcal{I}$ is a model of $\mathcal{K}$, this contradicts $a <A^{\mathcal{I}}$. Consider an $(x_A, x) \in \sigma$. We have that $x_A \in B^{\mathcal{G}_\mathcal{K}}$ if and only if $\mathcal{T} \models A \sqsubseteq B$ and by construction of σ we have that $x \in A^{\mathcal{I}}$. But since $\mathcal{I}$ is a model of $\mathcal{T}$ it follows that $x \in B^{\mathcal{I}}$.

For the [Forth] condition, we consider the three possible kinds of combinations of elements of $\text{ind}(\mathcal{A})$ and N in $r^{\mathcal{G}_\mathcal{K}}$. First, consider $(a, b) \in r^{\mathcal{G}_\mathcal{K}}$ with $a, b \in \text{ind}(\mathcal{A})$ and $(a, a) \in \sigma$. This means $r(a, b) \in \mathcal{A}$ by construction of $\mathcal{G}_\mathcal{K}$ and $(b, b) \in \sigma$. Since $\mathcal{I}$ is a

model of $\mathcal{A}$, it follows that $(a, b) \in r^{\mathcal{I}}$. Second, consider an $(a, x_A) \in r^{\mathcal{G}_\mathcal{K}}$ with $a \in \text{ind}(\mathcal{A})$, $x_A \in N$ and $(a, a) \in \sigma$. This means that $\mathcal{K} \models \exists r.A(a)$ by construction of $\mathcal{G}_\mathcal{K}$. Since $\mathcal{I}$ is a model of $\mathcal{K}$, there must be a $x \in A^{\mathcal{I}}$ such that $(a, x) \in r^{\mathcal{I}}$. By construction of σ, we have that $(x_A, x) \in \sigma$. Finally, consider an $(x_A, x_B) \in r^{\mathcal{G}_\mathcal{K}}$ with $x_A, x_B \in N$ and $(x_A, x) \in \sigma$. By construction of $r^{\mathcal{G}_\mathcal{K}}$ we have that $\mathcal{T} \models A \sqsubseteq \exists r.B$ and by construction of σ that $x \in A^{\mathcal{I}}$. Since $\mathcal{I}$ is a model of $\mathcal{K}$, there must be an $x' \in B^{\mathcal{I}}$ such that $(x, x') \in r^{\mathcal{I}}$ and it follows that $(x_B, x') \in \sigma$ by construction of σ. □

3.4 Concepts as Interpretations

In this section, we describe what it means to "view" an $\mathcal{EL}$ concept C as an interpretation $\mathcal{I}_C$ and how $\mathcal{I}_C$ relates to other interpretations. The tree-shaped concept interpretation $\mathcal{I}_C$ can be constructed by "viewing" C as an interpretation. Baader, Küsters, and Molitor [6] give a similar construction called $\mathcal{EL}$-description trees. Instead of constructing tree-shaped graphs, we construct tree-shaped interpretations (since we have defined simulations to be between interpretations, not graphs). Every $\mathcal{EL}$ concept C of depth 0 is of the form

$$C = A_1 \sqcap \cdots \sqcap A_n$$

and every concept C of depth $k + 1$ is of the form

$$C = A_1 \sqcap \cdots \sqcap A_n \sqcap \exists r_1.C_1 \sqcap \cdots \exists r_m.C_m$$

with each C_i being of depth at most k. Note that for $C = \top$ we consider $n = 0$. $\mathcal{I}_C$ can now be inductively constructed from this form as follows. First, if $\text{depth}(C) = 0$ then $\Delta^{\mathcal{I}_C} = \{v_C\}$ and $A_i^{\mathcal{I}_C} = \{v_C\}$ for all i (and all other $A^{\mathcal{I}_C}$ and $r^{\mathcal{I}_C}$ are empty). Then, If $\text{depth}(C) > 0$, let $\mathcal{I}_i$ be the inductively defined $\mathcal{I}_{C_i}$ for all $1 \leq i \leq m$. We consider the elements $\Delta^{\mathcal{I}_i}$ to be pairwise disjoint. Then

$$\begin{aligned}
\Delta^{\mathcal{I}_C} &= \{v_C\} \cup \bigcup_{1 \leq i \leq m} \Delta^{\mathcal{I}_i} \\
r^{\mathcal{I}_C} &= \bigcup_{1 \leq i \leq m} r^{\mathcal{I}_i} \cup \{(v_C, v_{C_i}) \mid \text{if } r_i = r\} \\
A^{\mathcal{I}_C} &= \bigcup_{1 \leq i \leq m} A^{\mathcal{I}_i} \cup \begin{cases} \{v_C\} & \text{if } A \in \{A_1, \ldots, A_n\} \\ \emptyset & \text{otherwise.} \end{cases}
\end{aligned}$$

for all r and A. Figure 3.3 shows the interpretation $\mathcal{I}_C$ for the concept $C = \text{Professor} \sqcap \exists\text{gives.Lecture} \sqcap \exists\text{attends.Lecture}$

The following lemma characterizes the semantic relation of $\mathcal{I}_C$ and C:

Lemma 3.12. *Let C be an $\mathcal{EL}$ concept, then the following holds for all interpretations $\mathcal{I}$ and elements $a \in \Delta^{\mathcal{I}}$:*

$$(\mathcal{I}_C, v_C) \preceq_\Sigma (\mathcal{I}, a) \text{ if and only if } \mathcal{I} \models C(a).$$

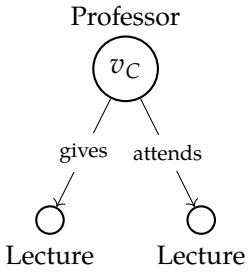


Figure 3.3: The tree-shaped interpretation $\mathcal{I}_C$ of C = Professor $\sqcap$ $\exists$gives.Lecture $\sqcap$ $\exists$attends.lecture

Proof. First, assume $(\mathcal{I}_C, v_C) \precsim (\mathcal{I}, a)$. We now show $\mathcal{I} \models C(a)$ by structural induction over C.

Case $C = A$: $v_C \in A^{\mathcal{I}_C}$ by construction of $\mathcal{I}_C$. Since $(\mathcal{I}_C, v_C) \precsim (\mathcal{I}, a)$, there is a simulation σ, such that $(v_C, a) \in \sigma$. Due to the first condition of simulations it follows that $a \in A^{\mathcal{I}}$ and thus $\mathcal{I} \models C(a)$.

Case $C = D \sqcap D'$: $v_C \in D^{\mathcal{I}_C} \cap D'^{\mathcal{I}_C}$ by construction of $\mathcal{I}_C$. Since the induction hypothesis holds for D and D' we have that $\mathcal{I} \models D(a)$ and $\mathcal{I} \models D'(a)$. Thus it follows that $a \in D^{\mathcal{I}} \cap D'^{\mathcal{I}}$ and $\mathcal{I} \models C(a)$.

Case $C = \exists r.D$: $v_C \in (\exists r.D)^{\mathcal{I}_C}$ by construction of $\mathcal{I}_C$. This means there is a $v'_C \in D^{\mathcal{I}_C}$ with $(v_C, v'_C) \in r^{\mathcal{I}_C}$. Since $(\mathcal{I}_C, v_C) \precsim (\mathcal{I}, a)$, there is a simulation σ, such that $(v_C, a) \in \sigma$. Due to the second condition of simulations there must be a $a' \in \Delta^{\mathcal{I}}$ with $(a, a') \in r^{\mathcal{I}}$ and $(v'_C, a') \in \sigma$. By applying the induction hypothesis we have that $a' \in D^{\mathcal{I}}$ and thus $\mathcal{I} \models \exists r.D(a)$.

Conversely assume that $\mathcal{I} \models C(a)$. We now show that there is a simulation σ witnessing $(\mathcal{I}_C, v_C) \precsim (\mathcal{I}, a)$ by structural induction over C.

Case $C = A$: We construct a simulation $\sigma = \{(v_C, a)\}$. By construction of $\mathcal{I}_C$ we know that $v_C \in A^{\mathcal{I}_C}$. Since $\mathcal{I} \models C(a)$ we have that $a \in A^{\mathcal{I}}$ and σ fulfills the [Atom] condition. Since v_C has no r-successors in $\mathcal{I}_C$, the relations σ fulfills the [Forth] condition and it follows that $(\mathcal{I}_C, v_C) \precsim (\mathcal{I}, a)$.

Case $C = D \sqcap D'$: Since the induction hypothesis holds for D and D', there are simulations σ_D and $\sigma_{D'}$ such that $(\mathcal{I}_D, v_C) \precsim (\mathcal{I}, a)$ and $(\mathcal{I}_{D'}, c) \precsim (\mathcal{I}, a)$. It follows that the simulation $\sigma = \sigma_D \cup \sigma_{D'}$ is a witness that $(\mathcal{I}_C, v_C) \precsim (\mathcal{I}, a)$.

Case $C = \exists r.D$: There is a $v'_C \in D^{\mathcal{I}_C}$ with $(v_C, v'_C) \in r^{\mathcal{I}_C}$ by construction of $\mathcal{I}_C$. Since $\mathcal{I} \models C(a)$ there must also be a $a' \in D^{\mathcal{I}}$ with $(a, a') \in r^{\mathcal{I}}$. The induction hypothesis holds for D, thus there is a simulation σ' such that $(\mathcal{I}_D, v'_C) \precsim (\mathcal{I}, a')$. We construct the simulation $\sigma = \sigma' \cup \{(v_C, a)\}$ which is a simulation between $(\mathcal{I}_C, v_C)$ and $(\mathcal{I}, a)$. □

Similarly we can "view" a tree-shaped interpretation $\mathcal{I}$ as a concept $C_{\mathcal{I}}$, by inductively constructing sub-concepts for all successors of an individual and adding them into a conjunction. This is conceptually the reverse of the construction described above and results for example in "reading of" the concept $C = \text{professor} \sqcap \exists\text{gives.lecture} \sqcap \exists\text{attends.lecture}$ from the interpretation in Figure 3.3.

4 Concept-by-Example for $\mathcal{EL}$

In this chapter, we investigate the concept-by-example problem. First, we define the problem of $\mathcal{EL}$ concept separability in $\mathcal{EL}$ knowledge bases to be the following decision problem CBE:

Input: $\mathcal{EL}$ knowledge base $\mathcal{K}$, a set of positive examples $P \subseteq \text{ind}(\mathcal{K})$, a set of negative examples $N \subseteq \text{ind}(\mathcal{K})$ and a signature Σ.

Question: Is there an $\mathcal{EL}$ concept C using only names from Σ such that

- $\mathcal{K} \models C(a)$ for all $a \in P$, and
- $\mathcal{K} \not\models C(b)$ for all $b \in N$?

We say that C is a *witness* or that it *witnesses* $(\mathcal{K}, P, N, \Sigma) \in$ CBE if it is such a concept.

As an example for the concept-by-example problem consider the $\mathcal{EL}$ knowledge base from the last chapter:

$$\mathcal{T} = \{\text{Student} \sqsubseteq \text{Human} \sqcap \exists\text{attends.Lecture}\}$$
$$\mathcal{A} = \{\text{Student(alice)}, \quad \text{Student(bob)}, \quad \text{Lecture(logic)},$$
$$\text{attends(bob, logic)}, \quad \exists\text{attends.Seminar(alice)}\}.$$

Given this knowledge base, the positive examples $P = \{\text{alice}, \text{bob}\}$ and the negative example $N = \{\text{logic}\}$ and a signature with all relevant names, an $\mathcal{EL}$ concept C such that $\mathcal{K} \models C(\text{alice})$ and $\mathcal{K} \models C(\text{bob})$ but $\mathcal{K} \not\models C(\text{logic})$ is Human or $\exists$attends.Lecture.

On the other hand there is no separating concept for $P = \{\text{bob}\}$ and $N = \{\text{alice}\}$, since for the concepts $C \in \{\top, \text{Human}, \text{Student}, \exists\text{attends.Lecture}\}$ we have that $\mathcal{K} \models C(\text{bob})$, but also that $\mathcal{K} \models C(\text{alice})$. The concept $C = \text{attends.Seminar}$ describes Alice but not Bob, but since $\mathcal{EL}$ lacks negation, it does not help us to get a concept that describes Bob but not Alice.

Note that another related problem, that is sometimes considered in the literature is concept *definability*. Here, the question is, if there is a concept query that exactly describes only the positive examples, but no other individuals. This is a special case of CBE, where $N = \text{ind}(\mathcal{K}) \setminus P$.

In the following chapter we give a characterization of CBE using the canonical model and use it to show upper and lower bounds for the decision and computation problem variant. Furthermore, we show, that the lower bound does not improve if we restrict the knowledge base in various ways, for example to an empty TBox or to a fixed outdegree.

In the following sections we can assume $\mathcal{K}$ to be in normal form:

Theorem 4.1. *Let $\mathcal{K}$ be a knowledge base, P a set of positive examples, N a set of negative examples and Σ a signature. Let $\widehat{\mathcal{K}}$ denote the knowledge base $\mathcal{K}$ in normal form.*

$$(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE} \text{ if and only if } (\widehat{\mathcal{K}}, P, N, \Sigma) \in \mathsf{CBE}$$

Note that this requires the assumption that the new concept names introduced by the normalization are not from Σ.

Proof. ($\Rightarrow$) For the first direction, assume that $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$. Then there is a concept C using only names from Σ such that for all models $\mathcal{I}$ of $\mathcal{K}$ we have that $\mathcal{I} \models C(a)$ for all $a \in P$ and $\mathcal{I} \not\models C(b)$ for all $b \in N$. Since $\widehat{\mathcal{K}}$ is a conservative extension of $\mathcal{K}$, every model of $\widehat{\mathcal{K}}$ is also a model of $\mathcal{K}$. By definition C contains only names from Σ, it follows that $\widehat{\mathcal{K}} \models C(a)$ for all $a \in P$ and $\widehat{\mathcal{K}} \not\models C(b)$ for all $b \in N$. Thus C is a witness for $(\widehat{\mathcal{K}}, P, N, \Sigma) \in \mathsf{CBE}$.

($\Leftarrow$) Assume that $(\widehat{\mathcal{K}}, P, N, \Sigma) \in \mathsf{CBE}$. Then there is a concept C using only names from Σ such that for all models $\mathcal{I}$ of $\widehat{\mathcal{K}}$ we have that $\mathcal{I} \models C(a)$ for all $a \in P$ and $\mathcal{I} \not\models C(b)$ for all $b \in N$. Since every model of $\mathcal{K}$ can be extended to a model of $\widehat{\mathcal{K}}$ and C only contains names from Σ, it follows that $\mathcal{K} \models C(a)$ for all $a \in P$ and $\mathcal{K} \not\models C(b)$ for all $b \in N$. Thus C is a witness for $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$. □

4.1 Characterization of Separability

In this section, we provide a characterization of the concept-by-example problem for $\mathcal{EL}$ knowledge bases and $\mathcal{EL}$ concepts. This characterization allows us to develop algorithms for both concept learning and deciding the separability problem in the following sections. The idea of this characterization is based on the characterization of the similar *query-by-example* problem for relational databases by ten Cate and Dalmau [39] and Barceló and Romero [9] and its extension to description logic knowledge bases by Gutiérrez-Basulto, Jung, and Sabellek [19]. The main differences to these earlier works are that we use $\mathcal{EL}$ concepts instead of conjunctive queries, finite canonical models instead of universal interpretations and Σ-simulations instead of Σ-homomorphisms. The following theorem states the characterization of separability.

Theorem 4.2. *For every $\mathcal{EL}$ knowledge base $\mathcal{K}$, all sets P and N over $ind(\mathcal{A})$ and signatures Σ, the following two are equivalent*

1. $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$
2. $\prod_{a \in P} (\mathcal{G}_\mathcal{K}, a) \not\preceq_\Sigma (\mathcal{G}_\mathcal{K}, b)$ *for all* $b \in N$

Proof. First, let $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$ with $P = \{a_1, \dots a_n\}$ and let C be an $\mathcal{EL}$-concept witnessing this using only names from Σ. We want to show that 2. holds. By universality of $\mathcal{G}_\mathcal{K}$ (Theorem 3.11) and Lemma 3.12 there are simulations σ_i such that $(\mathcal{I}_C, v_C) \preceq (\mathcal{G}_\mathcal{K}, a_i)$

for each positive example a_i. From these we construct σ as a relation between $\mathcal{I}_C$ and $\prod_{a\in P}(\mathcal{G}_\mathcal{K}, a)$ as follows:

$$\sigma = \{(a, (a_1, \ldots, a_n)) \mid (a, a_i) \in \sigma_i \text{ for all } i\}.$$

By construction σ is a simulation from $\mathcal{I}_C$ to $\prod_{a\in P} \mathcal{G}_\mathcal{K}$ and $(v_C, (a_1, \ldots, a_n)) \in \sigma$.

Assume for contradiction that 2. does not hold. Then there is a $b \in N$ such that there is a simulation ρ between $\prod_{a\in P}(\mathcal{G}_\mathcal{K}, a)$ and $(\mathcal{G}_\mathcal{K}, b)$. Composing σ and ρ yields a simulation that is a witness for $(\mathcal{I}_C, c) \precsim (\mathcal{G}_\mathcal{K}, b)$ due to transitivity of simulations and thus $\mathcal{K} \models C(b)$ (Lemma 3.12), a contradiction to $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$.

Conversely, let 2. be fulfilled. We want to show that there is a witness concept C for $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$. Let $(\mathcal{I}, a^*)$ be the Σ-restriction of $\prod_{a\in P}(\mathcal{G}_\mathcal{K}, a)$. The interpretation $\mathcal{I}$ cannot directly be viewed as a concept. Instead we consider its tree-shaped unravelling $\widehat{\mathcal{I}}$, which can be viewed as a (possibly infinite) $\mathcal{EL}$ concept $C = C_{\widehat{\mathcal{I}}}$. In particular consider a^* is at the "root" of the tree-shaped concept. By construction of C, $\widehat{\mathcal{I}} \models C(a^*)$. From Lemma 3.7 and the existence of projections from $\mathcal{I}$ to $\mathcal{G}_\mathcal{K}$ it follows that $\mathcal{G}_\mathcal{K} \models C(a_i)$. From 2. and universality of $\mathcal{G}_\mathcal{K}$, we have that $\mathcal{K} \models C(a)$ for all $a \in P$ and $\mathcal{K} \not\models C(b)$ for all $b \in N$. The issue that remains is that the unravelling $\widehat{\mathcal{I}}$ may be infinite and we cannot read off a finite concept C if this is the case.

If C is infinite, we show that there is a finite restriction of C which is a witness. Denote with $C_i, i \geq 0$, the restriction of C to role depth i. Parts of C of the form $\exists r.D$ at role depth i are simply replaced with $\top$ to achieve this. We have that C_i is finite for every i. For contradiction, we assume that, for every $i \geq 1$, C_i is not a witness. There is a simulation σ_i such that $(\mathcal{I}_{C_i}, v_{C_i}) \precsim (\mathcal{G}_\mathcal{K}, b_i)$ for some $b_i \in N$ (Lemma 3.12) for each $C - i$. Since N is finite, there must be some b such that $b = b_i$ for infinitely many i. Thus there are infinitely many simulations $\sigma_i, i \geq 1$ with $(v_C, b) \in \sigma_i$. From this sequence of simulations we construct a new sequence $\sigma'_i, i \geq 1$ such that

$$\text{for all } j \geq 1 \text{ and } i < j, \text{ we have that } \sigma'_i \subseteq \sigma'_j\,. \tag{$*$}$$

Start with setting $\sigma'_1 = \sigma_1$, satisfying the condition ($*$). To define σ'_j, assume that all σ'_k are defined for $0 < k < j$. Let $V = \Delta^{\mathcal{I}_{C_j}} \setminus \Delta^{\mathcal{I}_{C_{j-1}}}$ be all individuals in $\mathcal{I}_{C_j}$ but not in $\mathcal{I}_{C_{j-1}}$ and define for all $k \geq j$, τ_k as the restriction of σ_k to V. By construction, V is finite. Moreover as $\mathcal{G}_\mathcal{K}$ has finite outdegree, there are only finitely many different τ_k. Choose some τ such that $\tau = \tau_k$ for infinitely many $k \geq j$. Then obtain a new sequence of simulations by dropping all σ_k with $\tau_k \neq \tau$. Setting $\sigma'_j = \tau \cup \sigma'_{j-1}$ finishes the construction and satisfies ($*$). It remains to note that $\widehat{\sigma} = \bigcup_{i\geq 0} \sigma'_i$ is a simulation from $(\mathcal{I}_C, v_C)$ into $(\mathcal{G}_\mathcal{K}, b)$. Thus $\prod_{a\in P}(\mathcal{G}_\mathcal{K}, a) \precsim (\mathcal{G}_\mathcal{K}, b)$, contradicting 2., it follows that there must be an i such that C_i is a witness for CBE. □

4.2 Deciding Separability is ExpTime-complete

Now that we have this characterization of the concept-by-example problem for $\mathcal{EL}$ knowledge bases, we can use it to give an algorithm for deciding the problem and thus show an upper bound for its complexity.

Input: $(\mathcal{K}, P, N, \Sigma)$

1. Construct $\mathcal{G}_\mathcal{K}$ from $\mathcal{K}$.
2. Construct $\prod_{a \in P}(\mathcal{G}_\mathcal{K}, a)$ from $\mathcal{G}_\mathcal{K}$.
3. For all $b \in N$, check if $\prod_{a \in P}(\mathcal{G}_\mathcal{K}, a) \precsim_\Sigma (\mathcal{G}_\mathcal{K}, b)$. Return "false" if there exists such a simulation.
4. Otherwise return "true"

Figure 4.1: Algorithm for deciding CBE in exponential time

The algorithm in Figure 4.1 returns "true" if $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$ and "false" if $(\mathcal{K}, P, N, \Sigma) < \mathsf{CBE}$. Soundness and completeness of this algorithm follow directly from the characterization in Theorem 4.2. We claim that this algorithm runs in exponential time in the size of its input:

- Step 1 takes polynomial time to construct $\mathcal{G}_\mathcal{K}$ from $\mathcal{K}$.
- Step 2 constructs $\prod_{a \in P}(\mathcal{G}_\mathcal{K}, a)$ with exponential size in $|P|$ and thus requires exponential time.
- The check in Step 3 is executed at most $|N|$ times and each step checks if there is a simulation between the exponentially-sized $\prod_{a \in P}(\mathcal{G}_\mathcal{K})$ and $(\mathcal{G}_\mathcal{K}, b)$. This can be decided with the polynomial-time algorithm for simulations from Section 3.2. Overall, each check thus takes exponential time, since we need to check for the existence of a simulation between an exponentially-sized and a polynomially-sized interpretation.

Thus, the described algorithm can be implemented to run in exponential time in the size of its input. The lemma directly follows from this:

Lemma 4.3. CBE *is in ExpTime.*

For showing ExpTime-hardness of CBE and thus give a matching upper bound we adapt a proof of ExpTime-hardness of the simulation problem for concurrent transition systems by Harel, Kupferman, and Vardi [20]. More precisely, we reduce the word problem for alternating, linear space bounded Turing machines (TMs). Given such a TM M with linear space bound $s(n)$ and an input w with $|w| = n$, we construct an ABox $\mathcal{A}$ and sets of positive and negative examples P and N, such that

$$((\emptyset, \mathcal{A}), P, N, \Sigma) \in \mathsf{CBE} \text{ if and only if } M \text{ does not accept } w.$$

It is well-known that there is a fixed alternating TM whose word problem is ExpTime-complete [15]. This construction for CBE was already published by Funk et al. [18].

For our purposes, an alternating Turing machine $M = (\notin, Q_\forall, Q_\exists, \mapsto, q_0, F_{acc}, F_{rej})$ consists of

- a finite set of tape symbols $\notin$,
- a set of universal states $Q_\forall$,
- set of existential states $Q_\exists$,
- a set of accepting states F_{acc},
- a set of rejecting states F_{rej} (the sets $Q_\forall$, $Q_\exists$, F_{acc}, F_{rej} of states are disjoint, their union is the set of all states Q),
- an initial state q_0,
- a transition relation $\mapsto \subseteq Q \times \notin \times Q \times \notin \times \{L, R, H\}$.

L, R and H correspond to the head moving to the left, to the right and staying at the same cell, respectively. We call the accepting and rejecting states $F_{acc} \cup F_{rej}$ final states. In our model of alternation, $\mapsto$ has a branching degree of 2. The TM is in an existential state in even-numbered steps and in a universal state in odd-numbered steps. We use $(q, a) \mapsto ((q_l, b_l, \Delta_l), (q_r, b_r, \Delta_r))$ when M is in state $q \in Q_\forall \cup Q_\exists$ reading symbol a to indicate that it branches to the left with (q_l, b_l, Δ_l) and to the right with (q_r, b_r, Δ_r). These directions are not related to the movement of the head which is determined by Δ_l or Δ_r. We call q_l the $\swarrow$*-child* of q and q_r the $\searrow$*-child* of q. Additionally, we assume that M always reaches a final state and that it loops there forever.

The computation of M on an input word w can be represented as a graph, whose nodes are configurations of M. With each node in the graph we associate an acceptance value 1 or 0 as follows. Configurations that are in an accepting state have the acceptance value 1, configurations that are in a rejecting state have acceptance value 0. The acceptance value of a configuration in a universal state is the minimum value of its two children and the value of a configuration in an existential state is the maximum value of its children. A TM accepts its input if the initial configuration has an acceptance value of 1 and rejects its input if the initial configuration has an acceptance value of 0.

Given an alternating TM M and a word w, we construct an ABox $\mathcal{A}$ consisting of two parts $\mathcal{A}_{M,w}$ and $\mathcal{B}_M$. Let $M = (\notin, Q_\forall, Q_\exists, \mapsto, q_0, F_{acc}, F_{rej})$ be the given TM, and let $s(n)$ be the space bound of M on the word w as input. We use concept names Reject and Accept and role names $r_{q,a,d,i}$, for all $q \in Q$, $a \in \notin$, $d \in \{\swarrow, \searrow\}$ and $1 \leq i \leq s(n)$ in both parts of the ABox.

We start with $\mathcal{B}_M$. The individuals in $\mathcal{B}_M$ that correspond to universal and existential states are $(\forall, 0, 0, 0)$, $(\forall, 0, 1, 0)$, $(\forall, 1, 0, 0)$, $(\forall, 1, 1, 1)$. $(\exists, 0, 0, 0)$, $(\exists, 0, 1, 1)$, $(\exists, 1, 0, 1)$, $(\exists, 1, 1, 1)$. The intuition is that an element $(*, l, r, v)$ corresponds to a configuration of a TM with the following properties: its $\swarrow$-child has acceptance value l, its $\searrow$-child has acceptance value r and its own acceptance value therefore is v. Furthermore, there are two individuals that represent an acceptance or a rejection state of the TM. They are called 1 and 0, which corresponds to their acceptance value. The only concept assertions in $\mathcal{B}_M$ are $\mathsf{Reject}(0)$ and $\mathsf{Accept}(1)$.

We add the following role assertions to $\mathcal{B}_M$ in order to represent the transitions between configurations. For left branches, we have $r_{q,a,\swarrow,i}(e,e') \in \mathcal{B}_M$ for all $q \in Q, a \in \notin, 1 \leq i \leq s(n)$ and for $e = (*, l, r, v)$ if either $e' = (*', l', r', v')$, $*$ is the opposite type of state as $*'$ and $l = v'$, or alternatively $e' = l$. For right branches, we have $r_{q,a,\searrow,i}(e,e') \in \mathcal{B}_M$ for all $q \in Q, a \in \notin, 1 \leq i \leq n$, and for $e = (*, l, r, v)$ if either $e' = (*', l', r', v')$, $*$ is the opposite type of state as $*'$ and $r = v'$, or alternatively $e' = r$. We additionally, have $r(0,0), r(1,1) \in \mathcal{B}_M$ for all role names r.

The second part $\mathcal{A}_{M,w}$ has a number of concept and role assertions for each of the $s(n)$ tape cells. For every cell i we have individuals of the form (q,a,i) and (a,i), for all $q \in Q, a \in \notin$. An individual (a,i) represents that the content of cell i is a and the head of the TM is not on cell i. An individual (q,a,i) represents that the content of cell i is a, that the head of the TM is on cell i and that the TM is in state q. In the following description, the cases $i = 1$ and $i = s(n)$ are not treated in a special way, since we can assume that that M does not move its head beyond cell 1 or $s(n)$.

Informally, a role assertion $r_{q,a,d,i}(e,e')$ is included in $\mathcal{A}_{M,w}$ if in state q with the head at cell i and reading tape symbol a, M can change the tape cell represented by e to e' by taking a d-branch. Note that e and e' may be identical, meaning that the TM transition does not affect the tape cell.

More formally, each transition $(q,a) \mapsto ((q_l, b_l, \Delta_l), (q_r, b_r, \Delta_r))$ of M results in the following role assertions for each tape cell i in $\mathcal{A}_{M,w}$:

1. Role assertions that correspond to the head moving from cell i to cell $i-1$ or $i+1$. For each individual $(q,a,i) \in \text{ind}(\mathcal{A}_{M,w})$, we include:

$$r_{q,a,\swarrow,i}((q,a,i),(b_l,i)), \quad r_{q,a,\searrow,i}((q,a,i),(b_r,i))$$

2. Role assertions that correspond to the head moving from cell $i-1$ or $i+1$ to cell i. For each individual $(b,i) \in \text{ind}(\mathcal{A}_{M,w})$, we include:

$$\begin{aligned}
&r_{q,a,\swarrow,i-1}((b,i),(q_l,b,i)), \text{ if } \Delta_l = R\\
&r_{q,a,\searrow,i-1}((b,i),(q_r,b,i)), \text{ if } \Delta_r = R\\
&r_{q,a,\swarrow,i+1}((b,i),(q_l,b,i)), \text{ if } \Delta_l = L\\
&r_{q,a,\searrow,i+1}((b,i),(q_r,b,i)), \text{ if } \Delta_r = L
\end{aligned}$$

3. Role assertions that correspond to the transition not modifying the cell. For each $(b,i) \in \text{ind}(\mathcal{A}_{M,w})$, we include:

$$\begin{aligned}
&r_{q,a,\swarrow,j}((b,i),(b,i)), \text{ for all } j < \{i-1, i\} \text{ if } \Delta_l = R\\
&r_{q,a,\searrow,j}((b,i),(b,i)), \text{ for all } j < \{i-1, i\} \text{ if } \Delta_r = R\\
&r_{q,a,\swarrow,j}((b,i),(b,i)), \text{ for all } j < \{i, i+1\} \text{ if } \Delta_l = L\\
&r_{q,a,\searrow,j}((b,i),(b,i)), \text{ for all } j < \{i, i+1\} \text{ if } \Delta_r = L\\
&r_{q,a,\swarrow,j}((b,i),(b,i)), \text{ for all } j \neq i \text{ if } \Delta_l = H\\
&r_{q,a,\searrow,j}((b,i),(b,i)), \text{ for all } j \neq i \text{ if } \Delta_r = H
\end{aligned}$$

4. Role assertions that modify the current cell without moving the head. For each $(q, a, i) \in \mathrm{ind}(\mathcal{A}_{M,w})$, we include:

$$r_{q,a,\swarrow,i}((q,a,i),(q_l,b_l,i)), \text{ if } \Delta_l = H$$
$$r_{q,a,\searrow,i}((q,a,i),(q_r,b_r,i)), \text{ if } \Delta_r = H$$

Additionally, we need a number of role assertions for the final transitions $(q, a) \mapsto (q, a, H)$, $q \in F_{acc} \cup F_{rej}$ of M. For each such transition and each possible cell i, we include the assertions

$$r_{q,a,\swarrow,i}((q,a,i),(q,a,i)), \quad r_{q,a,\searrow,i}((q,a,i),(q,a,i)).$$

It remains to add concept assertions that mark accepting and rejecting states. We include, for all $a \in \notin, 1 \leq i \leq n$:

$$\mathsf{Reject}((q,a,i)) \in \mathcal{A}_{M,w}, \text{ for all } q \in F_{rej},$$
$$\mathsf{Reject}((a,i)) \in \mathcal{A}_{M,w},$$
$$\mathsf{Accept}((q,a,i)) \in \mathcal{A}_{M,w}, \text{ for all } q \in F_{acc},$$
$$\mathsf{Accept}((a,i)) \in \mathcal{A}_{M,w}.$$

This finishes the construction of $\mathcal{A}_{M,w}$. The signature Σ includes all used role and concept names. It remains to define P and N. Intuitively, P is a set of individuals representing the initial configuration of M on input w. Moreover, N consists of a single individual b. Formally, let w_i denote the i-th symbol of w and β be the symbol for an empty tape cell. We then define:

$$a_i = \begin{cases} (q_0, w_1, 1) & \text{if } i = 1 \\ (w_i, i) & \text{if } 2 \leq i \leq n \\ (\beta, i) & \text{if } n+1 \leq i \leq s(n) \end{cases}$$
$$P = \{a_1, \ldots, a_{s(n)}\}$$
$$b = (\forall, 1, 1, 1).$$

Lemma 4.4. $(\mathcal{K}, P, \{b\}, \Sigma) \in \mathsf{CBE}$ *for* $\mathcal{K} = (\emptyset, \mathcal{A}_{M,w} \cup \mathcal{B}_M)$ *if and only if* M *does not accept* w.

Proof. By Theorem 4.2 and the construction of the disjoint $\mathcal{A}_{M,w}$ and $\mathcal{B}_M$, it suffices to show that

$$M \text{ accepts } w \quad \text{if and only if} \quad \prod_{a \in P} (\mathcal{I}_{\mathcal{A}_{M,w}}, a) \preceq_\Sigma (\mathcal{I}_{\mathcal{B}_M}, b), \tag{$*$}$$

where, for any ABox $\mathcal{A}$, $\mathcal{I}_\mathcal{A}$ is the canonical model of the knowledge base that contains just $\mathcal{A}$. Before we give the formal proof, we provide some insight in the construction of $\mathcal{A}_{M,w}$. For this purpose, let us denote with $\mathcal{I}$ the product $\prod_{a \in P} \mathcal{I}_{\mathcal{A}_{M,w}}$. Moreover, for a configuration α of M, let x_α denote the element of $\Delta^\mathcal{I}$ corresponding to this configuration, where for a $(e_1, \ldots e_{s(n)}) \in \Delta^\mathcal{I}$, e_1 encodes the content of the first tape cell, $e_{s(n)}$ encodes the

content of the last tape cell and the single $e_i = (q, a, i)$ encodes the state and head position. We claim that elements of $\mathcal{I}$ that are reachable from $(a_1, \ldots, a_{s(n)})$ correspond precisely to the configurations in the computation of M on input w. Indeed, $(a_1, \ldots, a_{s(n)}) = x_{\alpha_0}$ for the initial configuration α_0 of M on input w. Moreover, we observe that, for any (non-final) configuration α of M, x_α has precisely two successors x_{α_l}, x_{α_r} in $\mathcal{I}$, where α_l, α_r are the successor configurations of α according to M's transition relation $\mapsto$. To see this, let q be the state, i be the head position and a be the current tape symbol in α. By construction of $\mathcal{A}_{M,w}$, the element (q, a, i) of x_α has precisely two successors, one for $r_{q,a,\swarrow,i}$ and one for $r_{q,a,\searrow,i}$. Furthermore, we have $(x_\alpha, x') \in r^{\mathcal{I}}_{q,a,\swarrow,i}$ if and only if $x' = x_{\alpha_l}$ is the $\swarrow$-child of x_α and $(x_\alpha, x') \in r^{\mathcal{I}}_{q,a,\searrow,i}$ if and only if $x' = x_{\alpha_r}$ is the $\searrow$-child of x_α. We proceed with the proof of $(*)$.

$(\Leftarrow)$ Suppose σ is a simulation between $\prod_{a \in P}(\mathcal{I}_{\mathcal{A}_{M,w}}, a)$ and $(\mathcal{I}_{\mathcal{B}_M}, b)$. The goal is to show that the initial configuration α_0 of M on input w is accepting. For proving this, we associate with every element $d \in \mathrm{ind}(\mathcal{B}_M)$ a value v_d by taking $v_d = v$ if d is of the shape $(*, l, r, v)$ and $v_d = d$ if $d \in \{0, 1\}$. Now, the desired statement is a consequence of the following claim.

Claim. For every configuration α reachable from α_0, we have that, if $(x_\alpha, d) \in \sigma$, then v_d is the acceptance value of α.

Proof of the Claim. We prove this by induction on the length of the longest path from α to a final configuration.

First, consider the case when α is a final configuration. By construction of $\mathcal{A}_{M,w}$, we have $x_\alpha \in \mathsf{Accept}^{\mathcal{I}}$ if α is accepting and $x_\alpha \in \mathsf{Reject}^{\mathcal{I}}$ if α is rejecting. By definition of $\mathcal{B}_M$ and $(x_\alpha, d) \in \sigma$, we know $d = 1$ or $d = 0$, respectively. Consider now the case that α is not a final configuration and $(x_\alpha, d) \in \sigma$. By definition of the role assertions in $\mathcal{B}_M$, we have that d is a universal (resp., existential) element $(\forall, *, *, *)$ (resp., $(\exists, *, *, *)$) if α is a universal (resp., existential) configuration. By what was said above, x_α has precisely two successors x_{α_l} and x_{α_r} in $\mathcal{I}$. By the simulation condition [Forth], we know that $(x_{\alpha_l}, d_l) \in \sigma$ and $(x_{\alpha_r}, d_r) \in \sigma$ for some elements d_l, d_r. By induction, we know that v_{d_l} and v_{d_r} are the acceptance values of α_l and α_r. By definition of $\mathcal{B}_M$, the acceptance value of α is v_d. This finishes the proof of the Claim.

$(\Rightarrow)$ Suppose M accepts the word w. Define a relation σ from elements x_α of $\mathcal{I}$ to individuals of $\mathcal{B}_M$. Let $(x_\alpha, v) \in \sigma$ for configurations α that are final with acceptance value v and $(x_\alpha, (*, l, r, v)) \in \sigma$ for configurations α if and only if $*$ is the type of α, l is the acceptance value of the $\swarrow$-child of α, r is the acceptance value of the $\searrow$-child of α, v is the acceptance value of α. We prove that σ is a simulation between $(\mathcal{I}, x_{\alpha_0})$ and $(\mathcal{I}_{\mathcal{B}_M}, b)$. For condition [Atom], we only have to consider the final elements that are in the extensions of Accept or Reject since there are no other concept names. The relation σ fulfills [Atom] since it relates final elements to final elements in $\mathcal{B}_M$. For [Forth], consider an element x_α with $(x_\alpha, d) \in \sigma$ for some non-final configuration α; the argument for final configurations is similar. By the remark above, we have to show that $(x_{\alpha_l}, d_l) \in \sigma$ and $(x_{\alpha_r}, d_r) \in \sigma$ for two elements d_l, d_r and the possible successor configurations α_l and α_r of α. But this is clear from the definition of σ. □

Note that this reduction can be modified to show ExpTime-hardness of $\mathcal{EL}$ concept definability as well.

This gives us the matching ExpTime lower bound (Lemma 4.4) to the ExpTime upper bound (Lemma 4.3) and we know that CBE is ExpTime-complete:

Theorem 4.5. CBE *is ExpTime-complete.*

This concludes this section about the concept-by-example decision problem.

4.3 Restrictions

As seen in this section, the deciding separability for concept-by-example for $\mathcal{EL}$-knowledge bases is surprisingly hard, considering that $\mathcal{EL}$ is an inexpressive description logic. We investigate various restrictions of the problem which often lead to lower complexity in other settings. However, we will discuss in this section that various restrictions of the decision problem still yield an ExpTime lower bound and thus do not help making the problem easier. One restriction however, improves the lower bound by limiting the role depth of the separating concept. This is discussed in Chapter 5.

Empty TBox First, one could consider the decision problem where the input knowledge base is restricted to have an empty TBox. However, as can be seen in Section 4.2, the reduction from alternating, linear space bounded Turing machines does not require a TBox. This variant is still ExpTime-hard. This means that $\mathcal{EL}$ concept separability in relational databases is ExpTime-hard as well.

Tree-shaped ABox Then, one could restrict the input knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ to have a tree shaped ABox $\mathcal{A}$, but allow arbitrary TBoxes. In this case, the reduction from alternating Turing machines no longer applies, since it produces a cyclical ABox. Still, a slight modification of the reduction gives an ExpTime lower bound for this restriction: Instead of putting the complete cyclical structure for each tape cell into the ABox, it only contains a single "root" individual for each cell. Instead, we add sufficient concept inclusions to the TBox to generate this structure for each tape cell in the construction of the canonical model. The rest of the reduction then works exactly the same, since the canonical model of this modified reduction is in some sense equivalent to the model of the original reduction. It can be noted however, that this modification requires a non-empty and circular TBox.

Restrict the outdegree of the ABox Finally, one could restrict the outdegree of the individuals in the ABox and the signature of the knowledge base. In the reduction from alternating Turing machines as described in Section 4.2, the number of used role names $r_{q,a,d,i}$ depends on the size of the input, since i ranges from 1 to the number of tape cells the Turing machine needs for a input word of size n. Otherwise the number of role names and the ABox role assertions that determine the outdegree can be seen as fixed, since there exists a certain fixed alternating Turing machine with an ExpTime-hard word problem that can be used in the reduction [15].

To remove this dependence of the signature on the length of w, we modify the reduction from alternating, linear space bound Turing machines as follows: Given a TM with space bound $s(n)$ and input word w with $|w| = n$, the reduction constructs elements for $s(n)$ tape cells. Recall, that a role assertion $r_{q,a,d,i}(e, e')$ encodes that a cell e changes to e' if the head is at position $1 \leq i \leq s(n)$, the state is q, the current symbol is a and it branches in direction d. The intuition is that a single role transition $r_{q,a,d,i}$ is replaced by a path of role transitions of i times $r'_{q,a,d,1}$ followed by $s(n) - i + 1$ times $r_{q',a,d,0}$. Since at each

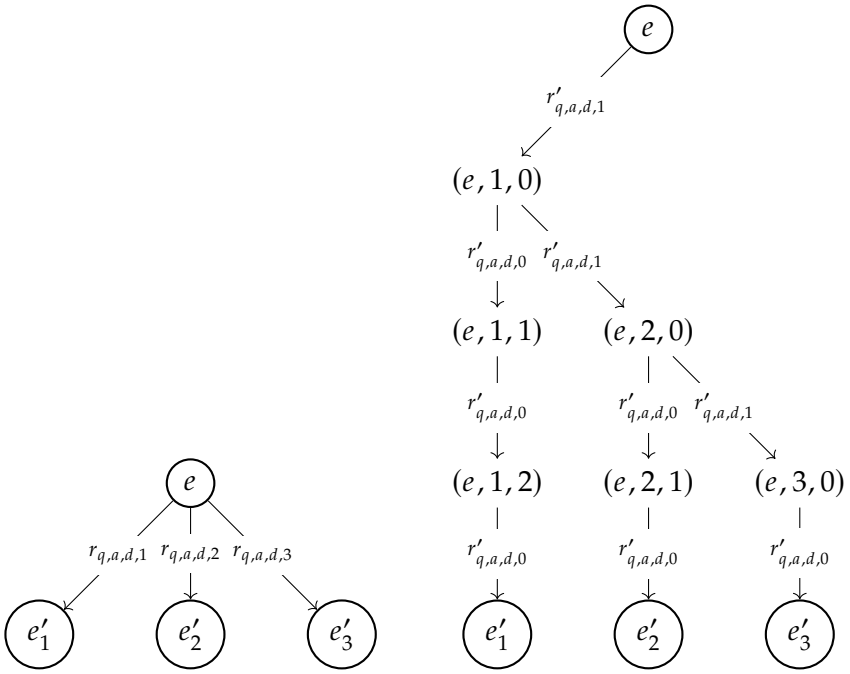


Figure 4.2: Illustration of the modification of the reduction

step in the path, the only options are $r_{q,a,d,0}$ and $r_{q,a,d,1}$, the outdegree of all individuals becomes 2. This replacement is done both in $\mathcal{A}_{M,w}$ and $\mathcal{B}_M$.

The modified reduction replaces the role assertion $r_{q,a,d,i}(e, e')$ with a sequence of additional elements $\{(e, j, 0) \mid 1 \leq j \leq s(n)\} \cup \{(e, i, k) \mid 1 \leq k < s(n) - i\}$, that are linked by the roles $r'_{q,a,d1}$ and $r'_{q,a,d,0}$ in the following way:

$$r'_{q,a,d,1}(e, (e, 1, 0))$$
$$r'_{q,a,d,1}((e, j, 0), (e, j + 1, 0)) \text{ for all } 1 \leq j < i$$
$$r'_{q,a,d,0}((e, i, k), (e, i, k + 1)) \text{ for all } 0 \leq k < s(n) - i - 1$$
$$r'_{q,a,d,0}((e, i, s(n) - i), e')$$

The cases $s(n) = 1$ or $i = s(n)$ or $i = (n) - 1$ require some special attention, but are not described here. This limits the outdegree of each individual to $q \times a \times d \times 2$ since the $(e, j, 0)$ are used in paths for multiple i.

Figure 4.2 shows the idea of this modification. The tape-size dependent outdegree of 3 in the first ABox is replaced by the fixed outdegree of at most 2 in the second ABox. Note that the path-length and number of additional elements of this construction can be improved if needed, however this does not change the idea behind the reduction.

With this modified reduction we have shown that bounding the outdegree of $\mathcal{K}$ still gives us an ExpTime lower bound and thus no improvement. This modification even

shows that we get the same lower bound for a knowledge base $\mathcal{K}$ over a fixed signature of concept and role names.

4.4 Size of the Learned Concept

Similarly to the decision problem, we can define a corresponding computation problem for the concept-by-example problem. Since it computes a concept that applies to all positive examples but not to the negative examples, it is often called *concept learning* in previous works. Here, we want to learn an $\mathcal{EL}$ concept that separates the examples in an $\mathcal{EL}$ knowledge base:

Input: $\mathcal{EL}$ knowledge base $\mathcal{K}$, a set of positive examples $P \subseteq \text{ind}(\mathcal{K})$, a set of negative examples $N \subseteq \text{ind}(\mathcal{K})$ and a signature Σ.

Output: An $\mathcal{EL}$ concept C using only names from Σ such that

- $\mathcal{K} \models C(a)$ for all $a \in P$, and
- $\mathcal{K} \not\models C(b)$ for all $b \in N$

if it exists and "fail" otherwise.

First we show that the learned concept C can be of double exponential size for some inputs. Second we show that there always is a concept C of at most double exponential size that witnesses $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$. We conclude that an algorithm that computes a separating concept must take double exponential time in the worst case.

Theorem 4.6. *For every $n \in \mathbb{N}$, there is an $\mathcal{EL}$ knowledge base $\mathcal{K}$, sets P and N and signature Σ such that $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$, the size of $(\mathcal{K}, P, N, \Sigma)$ is polynomial in n and the smallest separating concept C with*

- *$\mathcal{K} \models C(a)$ for all $a \in P$, and*
- *$\mathcal{K} \not\models C(b)$ for all $b \in N$*

has size of at least 2^{2^n}.

Proof. Let $n \in \mathbb{N}$. There is a $((\emptyset, \mathcal{A} \cup \mathcal{B}), P, \{b\}, \Sigma)$ such that the concept C that separates $a_1, \ldots a_n$ and b, is of double exponential size and the size of $\mathcal{A} \cup \mathcal{B}$ and P is polynomial in n.

We construct the first part $\mathcal{A}$ of the ABox such that the product model $\prod_{a \in P} \mathcal{G}_{\mathcal{K}}$ contains a bitwise counter with n bits as follows: For each bit $i \leq n$ we include elements 0_i and 1_i that represent the state of this bit (either 0 or 1). The concept name I marks the 1-state of the bit with the concept assertion $I(1_i)$. The possible state transitions of the bits are encoded with role assertions of roles r_i and l_i for $1 \leq i \leq n$. A r_i or l_i transition represents that the i-th bit of the counter flips from 0 to 1 and all lower bits flip from 1 to 0:

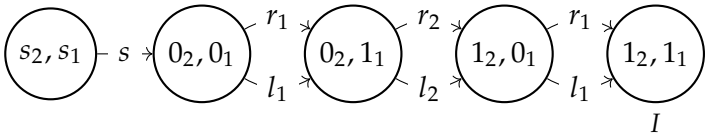


Figure 4.3: Product model of the $\mathcal{A}$ part of the ABox used in the reduction for $n = 2$ bits

$$r_i(0_i, 1_i), l_i(0_i, 1_i)$$
$$r_i(1_j, 0_j), l_i(1_j, 0_j) \text{ for all } 1 \leq j < i$$
$$r_i(1_j, 1_j), r_i(0_j, 0_j), l_i(1_j, 1_j), l_i(0_j, 0_j) \text{ for all } i < j \leq n$$

Additionally we include an element s_i for all $1 \leq i \leq n$ that represents an initial state for each bit and a special role s that represents a transition from the initial state to the 0-state with the role assertion $s(s_i, 0_i)$ for all $1 \leq i \leq n$. This completes $\mathcal{A}$.

We let P be $\{s_1, \ldots, s_n\}$, the initial states of the bits. Let $\mathcal{I}$ be $\prod_{a \in P} \mathcal{G}_{\mathcal{K}}$. Each reachable element in $\Delta^{\mathcal{I}}$ from $(s_1, \ldots, s_n)$ represents a binary number encoded in the bit states of the components, with the first component being the least significant bit and the last component being the most significant bit. The roles r_i and l_i link a number j to its successor $j + 1$ until the maximum value $2^n - 1$ of the counter is reached. The element representing the maximum value $(1_1, \ldots, 1_n)$ is the only element of $I^{\mathcal{I}}$. This way $\mathcal{I}$ has size 2^n and each counter element has two possible successors, one with an r_i role and one with an l_i role. Figure 4.3 shows $\mathcal{I}$ for the $\mathcal{A}$ part of the ABox for $n = 2$.

For the second part $\mathcal{B}$, we construct an ABox that encodes that there is a path over r_i and l_i that does not lead to an element that is in $I^{\mathcal{I}}$. We include an initial element t and elements that track acceptance values of paths $00, 10, 01$ and 11. The names of the elements $00, 10, 01$ and 11 contain the acceptance value of their left and right successor, the acceptance value of an element is 1 if both its l_i and r_i successors have acceptance value 1 and 0 otherwise. The acceptance value of element 11 is expressed with the concept assertion $I(11)$ The right and left successors are encoded with role assertions for all r_i and l_i:

$$l_i(11, 11), r_i(11, 11), l_i(10, 11), r_i(01, 11)$$
$$r_i(10, e), l_i(01, e), l_i(00, e), r_i(00, e) \text{ for all } e \in \{00, 10, 01\}$$

Additionally we add role assertions from the initial element t: $s(t, e)$,for all $e \in \{00, 10, 01\}$. This results in the ABox part shown in Figure 4.4.

It can now be observed that $(s_0, \ldots, s_n)$ and t can be separated by an $\mathcal{EL}$-concept, since for all simulations σ between the two parts of the ABox it can only be be that $(e, 11) \in \sigma$ but not that $(e, 10) < \sigma$, $(e, 01) < \sigma$ and $(e, 00) < \sigma$ for all elements of the binary counter e. And thus we have that $((s_0, \ldots, s_n), t) < \sigma$ since t does not have 11 as an s-successor.

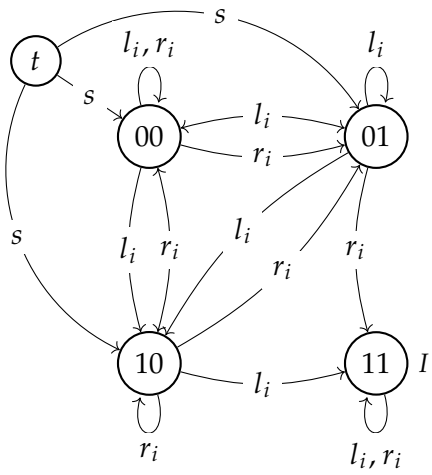


Figure 4.4: $\mathcal{B}$ part of the ABox used in the reduction

Claim: $C = \exists s.C_{2^n}$ is the smallest concept that separates $(s_0, \ldots, s_n)$ and t where

$$C_0 = I$$
$$C_{i+1} = \exists l_{f(i)}.C_i \sqcap \exists r_{f(i)}.C_i$$

and f "selects" the correct role name for the given depth[1]:

$$f(i) = \begin{cases} 0 & \text{if } i \text{ is even} \\ 1 + f(i/2) & \text{otherwise.} \end{cases}$$

Intuitively the $f(i)$-th bit is the bit that flips from 0 to 1 when a binary counter with the value i is incremented.

Proof of claim: Let $\mathcal{I}$ be the product interpretation $\prod_{a \in P} \mathcal{G}_{\mathcal{K}}$. To show that C is indeed the smallest concept, we enumerate the reachable elements of $\mathcal{I}$ according to their distance from the maximum counter value. The element $(1_1, \ldots, 1_n)$ with the maximum counter value $2^n - 1$ is then called v_0, its $r_{f(2^n-2)}$ and $l_{f(2^n-2)}$ predecessor is called v_1 and so on, until we reach the element $(0_1, \ldots, 0_n)$ with the minimum counter value 0, which is called v_{2^n-1}. We show that C_i is the smallest concept that separates v_i from 00, 01 and 10 by induction. For $i = 0$ we have that $v_0 \in I^{\mathcal{I}}$ but $01, 10, 00 < I^{\mathcal{I}}$. Thus, I is the smallest separating concept. Now assume that the claim holds for C_i. If we want to separate v_{i+1} from 00, 01 and 10, we need to separate both its $r_{f(2^n-i-2)}$ and its $l_{f(2^n-i-2)}$ successor from 00, 01, 10, since each of 00, 01, 10 has at least one successor in $\{00, 01, 10\}$. The smallest concept which does this is

$$C' = \exists r_{f(2^n-i-2)}.D_1 \sqcap l_{f(2^n-i-2)}.D_2$$

[1] Based on a possible definition of the integer sequence `https://oeis.org/A007814`

where D_1 separates the r_i successor and D_2 separates the l_i successor from 00, 01 and 10. Since both successors are v_i, the smallest separating concept is C_i by the induction hypothesis. It follows that

$$C' = C_{i+1}.$$

Thus, C_{i+1} is the smallest concept that separates v_{i+1} from 00, 10 and 01. It follows that $C = \exists s.C_{2^n}$ is the smallest concept that separates $(s_1, \dots, s_n)$ from t and

$$|C| \geq 2^{2^n}. \qquad \square$$

This gives us a double exponential lower bound on the size of the separating concept for concept-by-example. The next theorem gives us a matching upper bound:

Theorem 4.7. *Let $\mathcal{K}$ be an $\mathcal{EL}$ knowledge base, P a set of positive examples, N a set of negative examples and Σ a signature. If $(\mathcal{K}, P, N, \Sigma) \in$ CBE, then there is a concept of at most double exponential size witnessing this.*

Proof. To show this theorem we first prove a claim regarding the algorithm for simulations between finite interpretations from Section 3.2. Recall that it produces a sequence of relations σ_i until it ends with a simulation σ after at most n^2 steps.

Claim: Let $\mathcal{I}$ and $\mathcal{J}$ be interpretations and k be the maximum outdegree of $\mathcal{J}$. If the relation $\sigma_i \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{J}}$ does not contain a pair (c, d) then there is a concept C of size at most $(3k)^i$ such that $c \in C^{\mathcal{I}}$ but $d < C^{\mathcal{J}}$.

We will show this via induction over i. First, consider $i = 0$. If σ_0 does not contain a pair (c, d) then it must be that $c \in A$ but $d < A$ for some $A \in N_C$ ([Init]). The concept $C = A$ of size $(3k)^0 = 1$ thus separates c and d. Now, assume that the induction hypothesis holds for i. Let (c, d) be a pair that is in σ_i but removed by the algorithm in a [Step] and thus not in σ_{i+1}. Then there must be an $r \in N_R$ and $c' \in \Delta^{\mathcal{I}}$ with $(c, c') \in r^{\mathcal{I}}$, such that for no $d' \in \Delta^{\mathcal{J}}$ with $(d, d') \in r^{\mathcal{J}}$ we have that $(d', c') \in \sigma_i$. Via the induction hypothesis we know that c' can be separated from all r-successors $d_1, \dots, d_m$ of d. Note that m is bounded by the maximum outdegree of $\mathcal{J}$: $m \leq k$. Let C_i be the concept that separates c' from d_i, by the induction hypothesis we know that $|C_i| \leq (3k)^i$. The concept that separates c and d:

$$C = \exists r. \prod_{i=1}^{m} C_i$$

thus has a size of at most $(3k)^{i+1}$:

$$|C| \leq \underbrace{3}_{\exists r.} + \underbrace{k}_{\sqcap} + \underbrace{(3k)^i}_{C_i} \cdot k = 3 + k + 3^i \cdot k^{i+1} \leq (3k)^{i+1}.$$

From the characterization of concept-by-example in Theorem 4.2 we know that $(\mathcal{K}, P, N, \Sigma) \in$ CBE if and only if $\prod_{a \in P}(\mathcal{G}_{\mathcal{K}}, a) \not\preceq_{\Sigma} (\mathcal{G}_{\mathcal{K}}, b)$. Thus, running the algorithm for simulations on $\prod_{a \in P} \mathcal{G}_{\mathcal{K}}$ and $\mathcal{G}_{\mathcal{K}}$ does result in a relation that does not contain $(\prod_{a \in P} a, b)$ after at most an exponential number of steps (due to the exponential size of $\prod_{a \in P} \mathcal{G}_{\mathcal{K}}$).

Input: $(\mathcal{K}, P, N, \Sigma)$

1. Check with the algorithm for the decision problem if $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}$, output "false" if not.
2. Construct $\mathcal{G}_\mathcal{K}$ from $\mathcal{K}$.
3. Construct $\prod_{a \in P} \mathcal{G}_\mathcal{K}$ from $\mathcal{G}_\mathcal{K}$.
4. Construct $\mathcal{I}$ as the unravelling of $\prod_{a \in P}(\mathcal{G}_\mathcal{K})$ up to depth n^{2n} with $n = |(\mathcal{K}, P, N, \Sigma)|$.
5. Output $\mathcal{I}$ viewed as a concept C

Figure 4.5: Algorithm for concept learning in double exponential time

Let $n = |(\mathcal{K}, P, N, \Sigma)|$. We know that $|\mathcal{G}_\mathcal{K}| \leq n$, $\deg(\mathcal{G}_\mathcal{K}) \leq n$ and that $k \leq n$. Thus we have that $|\prod_{a \in P} \mathcal{G}_\mathcal{K}| \leq n^n$ and that the algorithm goes through at most n^{2n} steps. By the claim shown above we get the following upper bound for the size of the concept C that separates each a from each b:

$$|C| \leq (3n)^{n^{2n}}. \qquad \square$$

Theorem 4.7 and Theorem 4.6 tell us that a complete and correct algorithm for learning an $\mathcal{EL}$ concept in an $\mathcal{EL}$ knowledge base needs to run in double exponential time in the size of its input, if only to output the separating concept due to its size. Figure 4.5 gives a possible algorithm for this.

The algorithm runs in double exponential time in the size of its input:

- Step 1 can be implemented in exponential time as per Lemma 4.3.
- Steps 2 and 3 take exponential time as in the algorithm for the decision problem
- Since the outdegree of $\mathcal{G}_\mathcal{K}$ is bounded by n, we know that the size of the unravelling up to depth n^{2n} is at most $n^{n^{2n}}$. Step 4 thus takes a double exponential time. This cannot be avoided since the concept that we output from this is of the same size.
- Step 5 "reads of" the concept from the unravelling and thus requires double exponential time again.

That the concept C indeed separates P and N follows from the reasoning in the proof of the characterization in Theorem 4.2.

5 Concept-by-Example for $\mathcal{EL}$ with Bounded Concept Depth k

In the last chapter we have shown that deciding $\mathcal{EL}$ concept separability is ExpTime-complete in $\mathcal{EL}$ knowledge bases and that various restrictions do not change this. In this chapter we will define a variant of concept-by-example, that limits the role depth of the separating concept to a constant k, called CBE_k. We will show that CBE_k is only NP-complete and thus that this restriction makes the problem easier. Furthermore we will show that there is an exponential time algorithm for learning such a depth bounded separating concept.

Separability by an $\mathcal{EL}$ concept of at most role depth k or CBE_k is the following decision problem:

Input: $\mathcal{EL}$ knowledge base $\mathcal{K}$, a set of positive examples $P \subseteq \text{ind}(\mathcal{K})$, a set of negative examples $N \subseteq \text{ind}(\mathcal{K})$ and a signature Σ.

Question: Is there an $\mathcal{EL}$ concept C using only names from Σ and with $\text{depth}(C) \leq k$ such that

- $\mathcal{K} \models C(a)$ for all $a \in P$, and
- $\mathcal{K} \not\models C(b)$ for all $b \in N$?

We say that C is a *witness* or that it *witnesses* $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}_k$ if it is such a concept. As an example for the concept-by-example problem with concept role depth of at most one (CBE_1), consider the $\mathcal{EL}$ knowledge base from earlier:

$$\begin{aligned}\mathcal{T} =&\{\text{Student} \sqsubseteq \text{Human} \sqcap \exists\text{attends.Lecture}\}\\ \mathcal{A} =&\{\text{Student(alice)}, \quad \text{Student(bob)}, \quad \text{Lecture(logic)},\\ &\text{attends(bob, logic)}, \quad \exists\text{attends.Seminar(alice)}\}\end{aligned}$$

Given this knowledge base, the positive example $P = \{\text{alice}\}$ and the negative example $N = \{\text{bob}\}$ and a signature with all relevant names, the $\mathcal{EL}$ concept $C = \exists\text{attends.Seminar}$ with role depth 1 is a witness for $\mathcal{K} \models C(\text{alice})$ and $\mathcal{K} \not\models C(\text{bob})$. On the other hand, if we consider CBE_0, there is no concept C with $\text{depth}(C) = 0$ that separates $P = \{\text{alice}\}$ and $N = \{\text{bob}\}$ since for all such concepts $C \in \{\top, \text{Human}, \text{Student}\}$ it holds that $\mathcal{K} \models C(\text{alice})$ and $\mathcal{K} \models C(\text{bob})$. The individuals Alice and Bob can only be separated by a concept of role depth at least 1.

5.1 Characterization of Separability

To give a characterization of CBE_k we first need a notion of Σ-simulations for concepts that are limited to role depth k. These are Σ, k-simulations.

Definition 5.1. *Let $\mathcal{I}, \mathcal{J}$ be interpretations, $a \in \Delta^{\mathcal{I}}$ and $b \in \Delta^{\mathcal{J}}$.*

A relation σ_k is a Σ, k-simulation between $(\mathcal{I}, a)$ and $(\mathcal{J}, b)$ if there is a series of relations $\sigma_k \subseteq \sigma_{k-1} \subseteq \cdots \subseteq \sigma_0$ such that

1. *[AtomK] $c \in A^{\mathcal{I}} \wedge (c, d) \in \sigma_0 \implies d \in A^{\mathcal{J}}$ for all $A \in \Sigma$,*
2. *[ForthK] For all $r \in \Sigma$: If $(a, b) \in \sigma_{i+1}$ and $(a, a') \in r^{\mathcal{I}}$, then there exists $b' \in \Delta^{\mathcal{J}}$ with $(b, b') \in r^{\mathcal{J}}$ and $(a', b') \in \sigma_i$*
3. *[ElemK] $(a, b) \in \sigma_k$.*

We write $(\mathcal{I}, a) \preceq_{\Sigma,k} (\mathcal{J}, b)$ if there is a Σ, k-simulation between $(\mathcal{I}, a)$ and $(\mathcal{J}, b)$. Note that a Σ-simulation between $(\mathcal{I}, a)$ and $(\mathcal{J}, b)$ is also a Σ, k-simulation between $(\mathcal{I}, a)$ and $(\mathcal{J}, b)$ for any $k \geq 0$, since the [Forth] condition of Σ-simulations is stronger than the [ForthK] condition of Σ, k-simulations. Similarly to Theorem 3.4 there is a connection between Σ, k-simulations and concept with role depth of at most k, as expressed in the following lemma:

Lemma 5.2. *(Jung et al. [22]) Let $\mathcal{I}$ and $\mathcal{J}$ be interpretations. Then for all signatures Σ, all $\mathcal{EL}$ concepts C over names from Σ with depth$(C) \leq k$, all $c \in \Delta^{\mathcal{I}}$ and all $d \in \Delta^{\mathcal{J}}$, we have*

$$(\mathcal{I}, c) \preceq_{\Sigma,k} (\mathcal{J}, d) \text{ if and only if } c \in C^{\mathcal{I}} \implies d \in C^{\mathcal{J}}.$$

The proof of this lemma is standard and therefore not included here [32]. For the stronger k-bisimulations this proof can be found for the setting of modal logic in Blackburn, de Rijke, and Venema [10].

With this theorem we can give a characterization for the CBE_k problem. The characterization and its proof of correctness are similar to the characterization in Theorem 4.2 but use Σ, k-simulations instead of unbounded Σ-simulations. This makes handling the "infinite unravelling"-case in the proof simpler.

Theorem 5.3. *For every $\mathcal{EL}$ knowledge base $\mathcal{K}$, all sets P and N over ind$(\mathcal{A})$ and signatures Σ, the following are equivalent:*

1. $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}_k$
2. $\prod_{a \in P} (\mathcal{G}_{\mathcal{K}}, a) \not\preceq_{\Sigma,k} (\mathcal{G}_{\mathcal{K}}, b)$ *for all* $b \in N$

Proof. ($\Rightarrow$) Let C be the concept that witnesses $(\mathcal{K}, P, N, \Sigma) \in CBE_k$ with depth$(C) \leq k$. By Lemma 3.12 and since $\mathcal{G}_{\mathcal{K}}$ is a model of $\mathcal{K}$ there are Σ-simulations σ_i such that $(\mathcal{I}_C, v_C) \preceq_{\Sigma} (\mathcal{G}_{\mathcal{K}}, a_i)$, these are also Σ, k-simulations. From these we construct

$$\sigma = \{(a, (a_1, \ldots, a_n)) \mid (a, a_i) \in \sigma_i\}.$$

By construction, σ is a Σ, k-simulation between $(\mathcal{I}_C, v_C)$ and $\prod_{a \in P}(\mathcal{G}_\mathcal{K}, a)$ and we know that $(v_C, (a_1, \ldots, a_n)) \in \sigma$. Assume for contradiction that 2. does not hold. Then there is a $b \in N$ such that there is a Σ, k-simulation ρ between $\prod_{a \in P}(\mathcal{G}_\mathcal{K}, a)$ and $(\mathcal{G}_\mathcal{K}, b)$. Composing σ and ρ yields a Σ, k-simulation between $(\mathcal{I}_C, v_C)$ and $(\mathcal{G}_\mathcal{K}, b)$ and thus by Lemma 5.2 $\mathcal{K} \models C(b)$, a contradiction.

($\Leftarrow$) Assume now that $\prod_{a \in P}(\mathcal{G}_\mathcal{K}, a) \not\preceq_{\Sigma,k} (\mathcal{G}_\mathcal{K}, b)$ for all $b \in N$ is fulfilled. Let $(\widehat{\mathcal{I}}, a^*)$ be the tree-shaped unravelling up to depth k of $\prod_{a \in P}(\mathcal{G}_\mathcal{K}, a)$. We view $\widehat{\mathcal{I}}$ as an $\mathcal{EL}$-concept C with a^* as its root. By construction we have that $\widehat{\mathcal{I}} \models C(a^*)$ and thus it follows that $\mathcal{G}_\mathcal{K} \models C(a_i)$ for all i from Lemma 3.7 and the properties of the product model. From 2. and universality of $\mathcal{G}_\mathcal{K}$ (Theorem 3.11) we have that $\mathcal{K} \models C(a)$ for all $a \in P$ and $\mathcal{K} \not\models C(b)$ for all $b \in N$. □

5.2 Deciding Separability is NP-complete

In this section, we use the characterization of CBE_k to show that CBE_k is NP-complete. First, we have to show that the size of the witnessing concept is only polynomial in size of the input, instead of double exponential. We do this by showing that there always is an interpretation with a linear outdegree that can be used instead of the product of the canonical model, which can have an exponential outdegree. The combination of bounded role depth and linear outdegree then results in witness of polynomial size.

Let $\mathcal{I}$ and $\mathcal{I}'$ be interpretations. We call $\mathcal{I}'$ an *induced sub interpretation* of $\mathcal{I}$ if it is a restriction of $\mathcal{I}$ to a certain set of individuals, that is

$$\begin{aligned}
&\Delta^{\mathcal{I}'} \subseteq \Delta^{\mathcal{I}} \\
&A^{\mathcal{I}'} = A^{\mathcal{I}} \cap \Delta^{\mathcal{I}'} \text{ for all } A \in N_C \\
&r^{\mathcal{I}'} = r^{\mathcal{I}} \cap (\Delta^{\mathcal{I}'} \times \Delta^{\mathcal{I}'}) \text{ for all } r \in N_R,
\end{aligned}$$

Lemma 5.4. *Let $\mathcal{I}, \mathcal{J}$ be tree-shaped interpretations. Then for all $a \in \Delta^{\mathcal{I}}$ and $b \in \Delta^{\mathcal{J}}$, the following holds: If $(\mathcal{I}, a) \not\preceq_{\Sigma,k} (\mathcal{J}, b)$ then there is a sub-interpretation $\mathcal{I}'$ of $\mathcal{I}$ such that*

1. *$deg(\mathcal{I}') \leq deg(\mathcal{J})$,*
2. *$(\mathcal{I}', a) \not\preceq_{\Sigma,k} (\mathcal{J}, b)$.*

Proof. We show the lemma by induction on k. For $k = 0$, let $\mathcal{I}_{a,b,0}$ be the induced sub-interpretation of $\mathcal{I}$, defined by

$$\Delta^{\mathcal{I}_{a,b,0}} = \{a\}.$$

By construction $\deg(\mathcal{I}_{a,b,0}) \leq \deg(\mathcal{J})$ since the outdegree is 0. Furthermore, since there is no $\Sigma, 0$-simulation between $(\mathcal{I}, a)$ and $(\mathcal{J}, b)$, there must be an $A \in \Sigma$ such that $a \in A^{\mathcal{I}}$, but $b < A^{\mathcal{J}}$. Since $a \in A^{\mathcal{I}}_{a,b,0}$, there cannot be a $\Sigma, 0$-simulation between $(\mathcal{I}_{a,b,0}, a)$ and $(\mathcal{J}, b)$.

For $k > 0$, let a' be the r-successor of a in $\mathcal{I}$ and let $b_1, \dots b_n$ be the r-successors of b in $\mathcal{J}$ such that $(\mathcal{I}, a') \not\leq_{\Sigma,k-1} (\mathcal{J}, b_i)$ for all $i \leq n$. We construct a induced sub-interpretations $\mathcal{I}_{a,b,k}$ of $\mathcal{I}$, defined by

$$\Delta^{\mathcal{I}_{a,b,k}} = \{a\} \cup \bigcup_{b_i \in \{b_1,\dots,b_n\}} \Delta^{\mathcal{I}_{a',b_i,k-1}}$$

Let a' be an r-successor of a in $\mathcal{I}$. Since $\mathcal{I}$ is tree-shaped, the maximum outdegree of $\mathcal{I}_{a,b,k}$ is bounded by the number of r-successors of b in $\mathcal{J}$ and the outdegree of the sub-interpretations $\mathcal{I}_{a',b_i,k-1}$. By the induction hypothesis we thus know that

$$\deg(\mathcal{I}_{a,b,k}) \leq \deg(\mathcal{J}).$$

Furthermore for there to be a Σ, k-simulation between $(\mathcal{I}_{a,b,k}, a)$ and $(\mathcal{J}, b)$, there must be a $b' \in \Delta^{\mathcal{J}}$ such that $(b, b') \in r^{\mathcal{J}}$ and there is a $\Sigma, k-1$-simulation between $(\mathcal{I}_{a,b,k}, a')$ and $(\mathcal{J}, b)$. The element a' is contained in $\mathcal{I}_{a,b,k}$ since it is included in all $\mathcal{I}_{a',b_i,k-1}$. By the induction hypothesis and construction of $\mathcal{I}_{a,b,k}$ we know that there is no $\Sigma, k-1$-simulation between $(\mathcal{I}_{a',b_i,k-1}, a')$ and $(\mathcal{J}, b_i)$ for all b_i. Thus

$$(\mathcal{I}_{a,b,k}, a) \not\leq_{\Sigma,k} (\mathcal{J}, b). \qquad \square$$

With this bound on the outdegree of interpretations, we can show an upper bound of the size of the separating concept for CBE_k:

Lemma 5.5. *Let $\mathcal{K}$ be an $\mathcal{EL}$ knowledge base, P a set of positive examples, N a set of negative examples and Σ a signature. If $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}_k$, then there is a concept of at most polynomial size witnessing it.*

Proof. Assume that $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}_k$ with $|(\mathcal{K}, P, N, \Sigma)| = n$ and let the concept C be a witness for this. By the definition of CBE_k we know that $\mathrm{depth}(C) \leq k$. Since $\mathcal{K} \not\models C(b)$ for all $b \in N$, we know that

$$(\mathcal{I}_C, v_C) \not\leq_{\Sigma,k} (\mathcal{G}_{\mathcal{K}}, b) \text{ for all } b \in N.$$

First, consider a single $b_i \in N$. By Lemma 5.4 we know that there is a sub-interpretation $\mathcal{I}'_C$ of $\mathcal{I}_C$ with $\deg(\mathcal{I}'_C) \leq \deg(\mathcal{G}_{\mathcal{K}})$ and

$$(\mathcal{I}'_C, v_C) \not\leq_{\Sigma,k} (\mathcal{G}_{\mathcal{K}}, b_i).$$

Since $\deg(\mathcal{G}_{\mathcal{K}}) \leq n$ we know that $|\mathcal{I}'_C| \leq n^k$. If we view $\mathcal{I}'_C$ as a concept C_i we have that its size is polynomial in n. We still have $\mathcal{K} \models C_i(a)$ for all $a \in P$, since C_i is "created" from C by removing existential restrictions and thus $C^{\mathcal{J}} \subseteq C_i^{\mathcal{J}}$ must hold for all interpretations $\mathcal{J}$. Thus we have for the concept

$$C = \bigsqcap_{b_i \in N} C_i$$

that $\mathcal{K} \not\models C(b_i)$ for all $b_i \in N$ and that $\mathcal{K} \models C(a)$ for all $a \in P$ since $\mathcal{K} \models C_i(a)$ for all i. The concept C_i thus separates all $a \in P$ from all $b_i \in N$ and is of at most polynomial size in n. $\square$

Input: $(\mathcal{K}, P, N, \Sigma)$

1. Nondeterministically write a polynomially sized concept C on to the tape.
2. For all $a \in P$, check if $\mathcal{K} \models C(a)$. Reject the input if this is not the case.
3. For all $b \in N$, check if $\mathcal{K} \not\models C(b)$. Reject the input if this is not the case.
4. Otherwise accept the input.

Figure 5.1: Description of a NTM for deciding CBE_k

Lemma 5.6. $\mathsf{CBE}_k \in NP$ *for all* $k \geq 0$.

Proof. Figure 5.1 gives a description of a nondeterministic Turing machine (NTM) that decides CBE_k in polynomial time. Soundness and completeness follow from Lemma 5.5 and the definition of CBE_k.

- Step 1 only makes a polynomial amount of nondeterministic choices.
- Steps 2 and 3 check if $\mathcal{K} \models C(a)$ for a linear number of individuals a. Each check can be performed in polynomial time in the size of the concept and the knowledge base.

The NTM thus runs in nondeterministic polynomial time in the size of its input. It follows that $\mathsf{CBE}_k \in$ NP. □

We continue with showing a NP lower bound for CBE_k by giving a reduction from 3SAT. While we reduce to CBE_1, the reduction can be modified for any CBE_k, $k \geq 1$.

Lemma 5.7. CBE_k *for* $k \geq 1$ *is NP-hard.*

Proof. In order to show NP-hardness of the concept-by-example problem with fixed role depth k, we give a reduction from 3SAT. 3SAT is the Boolean satisfiability problem for formulas in conjunctive normal form where each clause is limited to three literals. It is well known that 3SAT is NP-complete [23]. A 3SAT formula φ is a conjunction of a number of clauses of the form $(l_1 \vee l_2 \vee l_3)$ where each literal l_i is either a variable x_i, or a negated variable $\overline{x_i}$. Let vars be the set of all variables. A formula φ is satisfiable if there is a variable assignment $v : \mathsf{vars} \to \{0, 1\}$ of the variables in φ that satisfies each clause, that is, evaluates at least one literal to 1.

For a given 3SAT formula φ we construct $\mathcal{A}, P, N, \Sigma$ such that $((\emptyset, \mathcal{A}), P, N, \Sigma) \in \mathsf{CBE}_1$ if and only if φ is satisfiable. The idea behind this reduction is based on the characterization of CBE_1 in Theorem 5.3. We construct a knowledge base $\mathcal{K}$ such that there is a $\Sigma, 1$-simulation between the product model and $\mathcal{G}_\mathcal{K}$ if for each variable assignment of φ there is a clause of φ that is not satisfied.

Given a propositional formula φ in conjunctive normal form, we construct two separate parts $\mathcal{A}_\varphi$ and $\mathcal{B}_\varphi$ of $\mathcal{A}$. We use a single role name r and the concept names X_i and $\overline{X}_i$ for each variable x_i of φ. Let S be the set of all such concept names

$$S = \{X_i \mid x_i \text{ is a variable in } \varphi\} \cup \{\overline{X}_i \mid x_i \text{ is a variable in } \varphi\}.$$

For the first part $\mathcal{A}_\varphi$ we add the elements a_i, p_i, n_i for each variable x_i in φ. We add the following assertions to $\mathcal{A}_\varphi$ for all x_i:

$$r(a_i, p_i) \quad r(a_i, n_i) \quad A(p_i) \text{ for all } A \in S \setminus \{\overline{X}_i\} \quad A(n_i) \text{ for all } A \in S \setminus \{X_i\}.$$

For the second part $\mathcal{B}_\varphi$, we consider the clauses of φ. $\mathcal{B}_\varphi$ contains a "root" element b and for each clause c_i of φ a child element b_i. For a given clause $c_i = (l_{i,1} \vee l_{i,2} \vee l_{i,3})$ we let $L_{i,j}$ be the concept name that corresponds to the variable of $l_{i,j}$ with the same polarity as $l_{i,j}$:

$$L_{i,j} = \begin{cases} X_k & \text{if } l_{i,j} = x_k \\ \overline{X}_k & \text{if } l_{i,j} = \overline{x_k} \end{cases}$$

The following assertions are added to $\mathcal{B}_\varphi$ for c_i:

$$r(b, b_i) \quad A(b_i) \text{ for all } A \in S \setminus \{L_{i,1}, L_{i,2}, L_{i,3}\}\}$$

This completes $\mathcal{B}_\varphi$. Now let

$$\begin{aligned} \mathcal{K} &= (\emptyset, \mathcal{A}_\varphi \cup \mathcal{B}_\varphi) \\ P &= \{a_i \mid x_i \in \mathsf{vars}(\varphi)\} \\ N &= \{b\} \\ \Sigma &= S \cup \{r\}. \end{aligned}$$

It remains to be shown that the reduction is sound and complete:

$$(\mathcal{K}, P, N, \Sigma) \in CBE_1 \text{ if and only if } \varphi \text{ is satisfiable.}$$

We show the contrapositive of this statement:

($\Rightarrow$) Assume that φ is not satisfiable. We need to show that $(\mathcal{K}, P, N, \Sigma) < CBE_1$. If φ is not satisfiable, then for all variable assignments v there must be a clause c_v that is not fulfilled. Let $b_v \in \Delta^{\mathcal{G}_\mathcal{K}}$ be the individual in $\mathcal{B}$ that corresponds to c_v.

Let $\mathcal{I}$ denote the product model $\prod_{a \in P}(\mathcal{G}_\mathcal{K}, a)$ and let $g_{v,i} \in \Delta^{\mathcal{G}_\mathcal{K}}$ be the individual that encodes the truth value of a variable x_i in the variable assignment v:

$$g_{v,i} = \begin{cases} p_i & \text{if } v(x_i) = 1 \\ n_i & \text{if } v(x_i) = 0 \end{cases}$$

For a variable assignment v, the individual $e_v = (g_{v,1}, \ldots, g_{v,n}) \in \Delta^\mathcal{I}$ corresponds to v by encoding the value of all variables in order.

Let σ be the following relation

$$\sigma = \{(e_v, b_v) \mid v \text{ is a variable assignment of } \varphi\} \cup \{((a_1, \ldots, a_n), b)\}$$

We want to show that σ is a Σ-simulation that witnesses

$$\prod_{a \in P} (\mathcal{G}_\mathcal{K}, a) \precsim_\Sigma (\mathcal{G}_\mathcal{K}, b). \tag{$*$}$$

For the [Atom] condition of simulations, assume that $e_v \in X_i^\mathcal{I}$ for some i and e_v. By construction of $\mathcal{A}$ and e_v we know that $v(x_i) = 1$. Since v does not fulfill the clause c_v, the variable x_i cannot appear as a positive literal in c_v. By construction of $\mathcal{B}$ we thus know that $b_v \in X_i^\mathcal{I}$. The reasoning for $e_v \in \overline{X_i}^\mathcal{I}$ for some i is equivalent. The [Atom] condition holds for $((a_1, \ldots, a_n), b)$ since $(a_1, \ldots, a_n)$ is in the extension of no concept name.

The [Forth] condition of simulations holds for all e_v since they have no r-successors. It remains to be shown that all r-successors of $(a_1, \ldots, a_n)$ are of the form e_v and are thus included in σ. Each successor must contain either p_i or n_i at position i, since these are the only successors of a_i. Thus σ is a Σ-simulation that witnesses $(*)$. It follows that $(\mathcal{K}, P, N, \Sigma) < CBE_1$ since σ is also a $\Sigma, 1$-simulation.

($\Leftarrow$) Assume that $(\mathcal{K}, P, N, \Sigma) < CBE_1$. By Theorem 5.3 there must be a $\Sigma, 1$-simulation σ between $(\mathcal{I}, (a_1, \ldots, a_n))$ and $(\mathcal{G}_\mathcal{K}, b)$ witnessing this. Given this $\Sigma, 1$-simulation we want to show that φ is not satisfiable by showing that for each variable assignment v there is a clause c_v of φ that is not fulfilled by that assignment. The individuals $(a_1, \ldots, a_n)$ and b must be included in the simulation σ. Since by construction e_v is an r-successor of $(a_1, \ldots, a_n)$ in $\mathcal{I}$, there must be an element c_i in $\mathcal{G}_\mathcal{K}$ that is a r-successor of b and $(e_v, c_i) \in \sigma$. Since e_v and c_i are included in a simulation, it follows from the [Atom] condition that for all concept names A, $e_v \in A^\mathcal{I} \implies c_i \in A^{\mathcal{G}_\mathcal{K}}$. We also know that by construction $c_i < A^{\mathcal{G}_\mathcal{K}}$ if A is a concept name that represents a variable truth-value that would make a literal of c_i true. Thus the variable assignment v does not make any literal of c_i true. It follows that for each variable assignment v, there is a clause c_v that is not fulfilled. There is no variable assignment that can satisfy all clauses. Therefore φ is not satisfiable. □

Theorem 5.8. CBE_k *is NP-complete for all* $k \geq 1$.

Proof. Follows directly from Lemma 5.6 and Lemma 5.7. □

Furthermore we can show that CBE_0 is in P. It is thus likely easier than CBE_k for $k \geq 1$. This is not surprising, since it does not allow any existential restrictions in the separating concept.

Theorem 5.9. $\mathsf{CBE}_0 \in P$.

Proof. Figure 5.2 shows an algorithm that decides CBE_0 in polynomial time in the size of its input. Correctness and soundness of the algorithm follow from the characterization of CBE_k in Theorem 5.3 and the fact that we only need to check the [Atomk] condition for $\Sigma, 0$-simulations.

Input: $(\mathcal{K}, P, N, \Sigma)$

1. Construct $\mathcal{G}_\mathcal{K}$
2. Construct only the product individual $(a_1, \ldots, a_n) \in \Delta^\mathcal{I}$ for $P = \{a_1, \ldots, a_n\}$ of the product model $\mathcal{I} = \prod_{a \in P} \mathcal{G}_\mathcal{K}$
3. For all $b \in N$, check if there is a concept name $A \in \Sigma$ with $(a_1, \ldots, a_n) \in A^\mathcal{I}$ and $b < A^{\mathcal{G}_\mathcal{K}}$
4. Accept if there is such an A for each b, Reject the input otherwise

Figure 5.2: Algorithm for deciding CBE_0 in polynomial time

Each step runs in polynomial time: Step 2 can be implemented in polynomial time, since we do not need to look at the entire product model, only a single element of it. The element can only be in the extension of a linear number of concept names. Step 3 runs in polynomial time since the size of N and Σ are bounded by the size of the input. □

5.3 Learning Role Depth Bounded Concepts

Similar to the case with unbounded role depth, we can define the corresponding concept learning problem for CBE_k:

Input: $\mathcal{EL}$ knowledge base $\mathcal{K}$, a set of positive examples $P \subseteq \text{ind}(\mathcal{K})$, a set of negative examples $N \subseteq \text{ind}(\mathcal{A})$ and a signature Σ.

Output: An $\mathcal{EL}$ concept C using only names from Σ and with $\text{depth}(C) \leq k$ such that

- $\mathcal{K} \models C(a)$ for all $a \in P$, and
- $\mathcal{K} \not\models C(b)$ for all $b \in N$

or "fail" if there is no such concept.

Lemma 5.5 tells us that if $(\mathcal{K}, P, N, \Sigma) \in \mathsf{CBE}_k$ for an $\mathcal{EL}$ knowledge base $\mathcal{K}$, a set of positive examples P, a set of negative examples N and a signature Σ, that there is a concept C of at most polynomial size such that

- $\mathcal{K} \models C(a)$ for all $a \in P$
- $\mathcal{K} \not\models C(b)$ for all $b \in N$.

This directly leads to an algorithm that enumerates all polynomially sized concepts and checks if they fulfill the conditions. Each check only takes a polynomial amount of time,

but there are exponentially many concepts of polynomial size. The algorithm thus would take exponential time in the size of its input.

If there exists an exponentially faster algorithm for learning the separating concept, that is an algorithm that runs in polynomial time, we could decide CBE_k in polynomial time by accepting the input if this algorithm outputs a concept and rejecting otherwise. This would show P = NP and is out of scope of this thesis. This differentiates the role depth bounded case from the unbounded case, where computing the separating concept is exponentially harder than deciding that it exists, as shown in Section 4.4.

6 Conclusion and Future Questions

In this thesis we have given a formal understanding of when a separating $\mathcal{EL}$ concept exists for a set of positive and a set of negative examples in an $\mathcal{EL}$ knowledge base. Deciding if such a concept exists is ExpTime-complete, however computing or learning a separating concept can take double exponential time in general. Some commonly considered restrictions do not improve this complexity. The exception is restricting the role depth of the separating concept, which makes deciding if such a concept exists NP-complete and allows for an exponential time algorithm to learn a separating concept. The algorithm is optimal if the exponential time hypothesis is true. Furthermore we show that the separating concept can be double exponentially sized in the unbounded case and only polynomially sized in the role depth bounded case.

Although there exist complete algorithms and implementations to learn $\mathcal{EL}$ concepts from positive and negative examples, no formal analysis of the existence of a separating $\mathcal{EL}$ concept or the complexity of $\mathcal{EL}$ concept learning has been done in previous work (except the work of Funk et al. [18], which considers separability in more expressive description logics beyond the results of this thesis).

These algorithms for concept-by-example help users understand and work with $\mathcal{EL}$ knowledge bases by allowing them to express concepts from the application domain with only positive and negative examples. This reduces the need to have expert knowledge in knowledge base engineering for a lot of tasks.

The following areas of future questions questions could be considered, to build upon the results of this thesis:

Extension to more expressive description logics $\mathcal{EL}$ is, although it has applications in real world knowledge bases, an inexpressive description logic. There are multiple extensions of $\mathcal{EL}$ that add expressiveness and mostly preserve the important property that subsumption and other common reasoning tasks can be done in polynomial time. Examples are $\mathcal{EL}_\bot$ and $\mathcal{EL}^{++}$ [4], although more exist.

Can the characterizations and algorithms be adapted to these extensions such that we get similar upper bounds?

Short representation of the separating concept There are $\mathcal{EL}$ knowledge bases for which concept learning takes double exponential time, primarily because a separating concept may be of double exponential size. The concept used in Section 4.4 however has a very regular structure and we conjecture that it could be represented with a set of TBox axioms of only single exponential size. Does learning a set of TBox axioms instead of a single concept make concept-by-example easier? Do the same lower bounds still apply?

Assumptions that lead to concept learning in polynomial time? Can we make other assumptions or restrictions about the knowledge base that would lead to polynomial time algorithms for deciding separability and concept learning? Are these assumptions and restrictions realistic in real world $\mathcal{EL}$ knowledge bases?

Apply techniques for better understanding of the least common subsumer The techniques used to characterize concept separability could be used to better understand the least common subsumer and most specific concept operators.

Role depth bounded case for $\mathcal{ELI}$ separability Funk et al. [18] have shown that $\mathcal{ELI}$ separability in the general case is undecidable. Considering that $\mathcal{EL}$ separability is considerably easier when one restricts the role depth of the separating concept, investigating if the lower bound of $\mathcal{ELI}$ separability improves for the same restriction, might be worthwhile.

Bibliography

[1] Marcelo Arenas and Gonzalo I. Diaz. "The Exact Complexity of the First-Order Logic Definability Problem". In: *ACM Transactions on Database Systems* 41.2 (2016), 13:1–13:14.

[2] Marcelo Arenas, Gonzalo I. Diaz, and Egor V. Kostylev. "Reverse Engineering SPARQL Queries". In: *Proceedings of the 25th International Conference on World Wide Web*. WWW 2016. Ed. by Jacqueline Bourdeau, Jim Hendler, Roger Nkambou, Ian Horrocks, and Ben Y. Zhao. 2016, pp. 239–249.

[3] Franz Baader. "Computing the Least Common Subsumer in the Description Logic EL w.r.t. Terminological Cycles with Descriptive Semantics". In: *Conceptual Structures for Knowledge Creation and Communication*. 11th International Conference on Conceptual Structures. Ed. by Aldo de Moor, Wilfried Lex, and Bernhard Ganter. 2003, pp. 117–130.

[4] Franz Baader, Sebastian Brandt, and Carsten Lutz. "Pushing the EL Envelope". In: *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*. Ed. by Leslie Pack Kaelbling and Alessandro Saffiotti. 2005, pp. 364–369.

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

[6] Franz Baader, Ralf Küsters, and Ralf Molitor. "Computing Least Common Subsumers in Description Logics with Existential Restrictions". In: *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*. IJCAI 99. Ed. by Thomas Dean. 1999, pp. 96–101.

[7] Franz Baader, Carsten Lutz, and Boontawee Suntisrivaraporn. "CEL—a Polynomial-Time Reasoner for Life Science Ontologies". In: *Proceedings of the 3rd International Joint Conference on Automated Reasoning*. IJCAR 2006. Ed. by Ulrich Furbach and Natarajan Shankar. Vol. 4130. Lecture Notes in Computer Science. 2006, pp. 287–291.

[8] Liviu Badea and Shan-Hwei Nienhuys-Cheng. "A Refinement Operator for Description Logics". In: *Inductive Logic Programming, 10th International Conference*. ILP 2000. Ed. by James Cussens and Alan M. Frisch. Vol. 1866. Lecture Notes in Computer Science. Springer, 2000, pp. 40–59.

[9] Pablo Barceló and Miguel Romero. "The Complexity of Reverse Engineering Problems for Conjunctive Queries". In: *20th International Conference on Database Theory*. ICDT 2017. Ed. by Michael Benedikt and Giorgio Orsi. Vol. 68. LIPIcs. 2017, 7:1–7:17.

[10] Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*. 2001.

[11] Angela Bonifati, Radu Ciucanu, and Aurélien Lemay. "Learning Path Queries on Graph Databases". In: *Proceedings of the 18th International Conference on Extending Database Technology*. EDBT 2015. Ed. by Gustavo Alonso et al. 2015, pp. 109–120.

[12] Sebastian Brandt. "On Subsumption and Instance Problem in ELH w.r.t. General TBoxes". In: *Proceedings of the 2004 International Workshop on Description Logics*. DL 2004. Ed. by Volker Haarslev and Ralf Möller. Vol. 104. CEUR Workshop Proceedings. 2004, pp. 21–30.

[13] Sebastian Brandt. "Polynomial Time Reasoning in a Description Logic with Existential Restrictions, GCI Axioms, and - What Else?" In: *Proceedings of the 16th Eureopean Conference on Artificial Intelligence*. ECAI 2004. Ed. by Ramó López de Mántaras and Lorenza Saitta. IOS Press, 2004, pp. 298–302.

[14] Lorenz Bühmann, Jens Lehmann, and Patrick Westphal. "DL-Learner - A Framework for Inductive Learning on the Semantic Web". In: *Journal of Web Semantics* 39 (2016), pp. 15–24.

[15] Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. "Alternation". In: *Journal of the Association of Computing Machinery* 28.1 (1981), pp. 114–133.

[16] Edmund M. Clarke and Bernd-Holger Schlingloff. "Model Checking". In: *Handbook of Automated Reasoning*. Ed. by John Alan Robinson and Andrei Voronkov. Elsevier and MIT Press, 2001, pp. 1635–1790.

[17] Nicola Fanizzi, Claudia d'Amato, and Floriana Esposito. "DL-FOIL Concept Learning in Description Logics". In: *Inductive Logic Programming, 18th International Conference*. ILP 2008. Ed. by Filip Zelezný and Nada Lavrac. Vol. 5194. Lecture Notes in Computer Science. Springer, 2008, pp. 107–121.

[18] Maurice Funk, Jean Christoph Jung, Carsten Lutz, Hadrien Pulcini, and Frank Wolter. "Learning Description Logic Concepts: When Can Positive and Negative Examples Be Separated?" In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. IJCAI 2019. (to appear). 2019.

[19] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Leif Sabellek. "Reverse Engineering Queries in Ontology-Enriched Systems: The Case of Expressive Horn Description Logic Ontologies". In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. IJCAI 2018. Ed. by Jérôme Lang. 2018, pp. 1847–1853.

[20] David Harel, Orna Kupferman, and Moshe Y. Vardi. "On the Complexity of Verifying Concurrent Transition Systems". In: *Information and Computation* 173.2 (2002), pp. 143–161.

[21] Pascal Hitzler, Markus Krotzsch, and Sebastian Rudolph. *Foundations of Semantic Web Technologies*. Chapman and Hall/CRC, 2009.

[22] Jean Christoph Jung, Fabio Papacchini, Frank Wolter, and Michael Zakharyaschev. "Model Comparison Games for Horn Description Logics". In: *Thirty-Fourth Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS. (to appear). 2019.

[23] Richard M. Karp. "Reducibility among Combinatorial Problems". In: *Complexity of Computer Computations*. Ed. by Raymond E. Miller, James W. Thatcher, and Jean D. Bohlinger. Springer US, 1972, pp. 85–103.

[24] Yevgeny Kazakov, Markus Krötzsch, and František Simančík. "The Incredible ELK - From Polynomial Procedures to Efficient Reasoning with EL Ontologies". In: *Journal of Automated Reasoning* 53.1 (2014), pp. 1–61.

[25] Boris Konev, Carsten Lutz, Ana Ozaki, and Frank Wolter. "Exact Learning of Lightweight Description Logic Ontologies". In: *Journal of Machine Learning Research* 18.201 (2018), pp. 1–63.

[26] Jens Lehmann. "DL-Learner: Learning Concepts in Description Logics". In: *Journal of Machine Learning Research* 10 (2009), pp. 2639–2642.

[27] Jens Lehmann. *Learning OWL Class Expressions*. Vol. 6. Studies on the Semantic Web. IOS Press, 2010.

[28] Jens Lehmann and Christoph Haase. "Ideal Downward Refinement in the EL Description Logic". In: *Inductive Logic Programming, 19th International Conference*. ILP 2009. Ed. by Luc De Raedt. Vol. 5989. Lecture Notes in Computer Science. Springer, 2009, pp. 73–87.

[29] Jens Lehmann and Pascal Hitzler. "Foundations of Refinement Operators for Description Logics". In: *Inductive Logic Programming, 17th International Conference*. ILP 2007. Ed. by Hendrik Blockeel, Jan Ramon, Jude W. Shavlik, and Prasad Tadepalli. Vol. 4894. Lecture Notes in Computer Science. Springer, 2007, pp. 161–174.

[30] Carsten Lutz, David Toman, and Frank Wolter. "Conjunctive Query Answering in the Description Logic EL Using a Relational Database System." In: *Proceedings of the 21st International Joint Conference on Artificial Intelligence*. IJCAI 2009. Ed. by Craig Boutilier. 2009, pp. 2070–2075.

[31] Carsten Lutz and Frank Wolter. "Conservative Extensions in the Lightweight Description Logic EL". In: *Proceedings of the 21th Conference on Automated Deduction*. CADE-21. Ed. by Franz Pfenning. Vol. 4603. Lecture Notes in Computer Science. Springer, 2007, pp. 84–99.

[32] Carsten Lutz and Frank Wolter. "Deciding Inseparability and Conservative Extensions in the Description Logic EL". In: *Journal of Symbolic Computation* 45.2 (2010), pp. 194–228.

[33] Shan-Hwei Nienhuys-Cheng and Ronald De Wolf. *Foundations of Inductive Logic Programming*. Vol. 1228. Springer Science & Business Media, 1997.

[34] Alan Rector and Ian Horrocks. "Experience Building a Large, Re-Usable Medical Ontology Using a Description Logic with Transitivity and Concept Inclusions". In: *Proceedings of the Workshop on Ontological Engineering*. AAAI'97. 1997, pp. 321–325.

[35] Peter N. Robinson and Sebastian Bauer. *Introduction to Bio-Ontologies*. Chapman and Hall/CRC, 2011.

[36] Md. Kamruzzaman Sarker and Pascal Hitzler. "Efficient Concept Induction for Description Logics". In: *CoRR* abs/1812.03243 (2018). arXiv: 1812.03243.

[37] Viachaslau Sazonau. "General Terminology Induction in Description Logics". PhD thesis. University of Manchester, 2017.

[38] Stefan Schulz, Ronald Cornet, and Kent Spackman. "Consolidating SNOMED CT's Ontological Commitment". In: *Applied ontology* 6.1 (2011), pp. 1–11.

[39] Balder ten Cate and Victor Dalmau. "The Product Homomorphism Problem and Applications". In: *18th International Conference on Database Theory*. ICDT 2015. Ed. by Marcelo Arenas and Martín Ugarte. Vol. 31. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015, pp. 161–176.

[40] Benjamin Zarrieß and Anni-Yasmin Turhan. "Most Specific Generalizations w.r.t. General EL-TBoxes". In: *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*. IJCAI 2013. 2013, pp. 1191–1197.

[41] Moshé M. Zloof. "Query-by-Example: The Invocation and Definition of Tables and Forms". In: *Proceedings of the International Conference on Very Large Data Bases*. Ed. by Douglas S. Kerr. ACM, 1975, pp. 1–24.