COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

OPTIMIZATION OF THE MHS-MXP ALGORITHM

Master thesis

2022

Bc. Janka Boborová

COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

OPTIMIZATION OF THE MHS-MXP ALGORITHM

Master thesis

Study Program:	Applied Informatics
Field of Study:	2511 Applied Informatics
School Department:	Department of Applied Informatics
Supervisor:	Mgr. Júlia Pukancová, PhD.
Consultant:	doc. RNDr. Martin Homola, PhD.

Bratislava, 2022 Bc. Janka Boborová

23185800

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname:	Bc. Janka Boborová
Study programme:	Applied Computer Science (Single degree study, master II. deg., full time form)
Field of Study:	Computer Science
Type of Thesis:	Diploma Thesis
Language of Thesis:	English
Secondary language:	Slovak

Title: Optimization of the MHS-MXP algorithm

Annotation: The MHS-MXP abduction algorithm improves MHS by applying a divide-and-conquer strategy (MXP) to search through a space of possible explanations (MHS-tree). The MXP runs are iterated, therefore suitable search heuristics, tree-pruning, and cached information from previous runs may possibly further optimize the search strategy in the consecutive iterations.

Aim: Propose improvements in MHS-MXP search strategy, and evaluate their efficiency on a suitable test case.

Literature: 1. Elsenbroich, C., Kutz, O., Sattler, U., 2006. A case for abductive reasoning over ontologies. In: OWLED
2. Shchekotykhin, K., Jannach, D., Schmitz, T., 2015. MergeXplain: Fast computation of multiple conflicts for diagnosis. In IJCAI
3. Homola, M., Pukancová, J., Gablíková, J., Fabianová, K., 2020. Merge, Explain, Iterate. In: DL

Supervisor:	Mgr. Júlia Pukancová, PhD.
Consultant:	doc. RNDr. Martin Homola, PhD.
Department:	FMFI.KAI - Department of Applied Informatics
Head of department:	prof. Ing. Igor Farkaš, Dr.
Assigned:	11.10.2021
Approved:	12.10.2021

prof. RNDr. Roman Ďurikovič, PhD.
Guarantor of Study Programme

.... Student

.... Supervisor

23185800

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta:	Bc. Janka Boborová
Študijný program:	aplikovaná informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor:	informatika
Typ záverečnej práce:	diplomová
Jazyk záverečnej práce:	anglický
Sekundárny jazyk:	slovenský

Názov: Optimization of the MHS-MXP algorithm
Optimalizácia algoritmu MHS-MXP

Anotácia: Abduktívny algoritmus MHS-MXP je zlepšením algoritmu MHS, ktoré využíva metódu rozdeľuj a panuj (MXP) pri prehľadávaní priestoru možných vysvetlení (MHS-strom). Behy MXP sú iterované, preto môže vhodná heuristika vyhľadávania, orezávanie stromu, a kešovanie informácií z predchádzajúcich behov potenciálne zlepšiť prehľadávaciu stratégiu v nasledujúcich iteráciách.

Cieľ: Navrhnúť možné zlepšenia prehľadávacej stratégie a evalvovať ich efektívnosť na vhodne zvolených testovacích dátach.

Literatúra: 1. Elsenbroich, C., Kutz, O., Sattler, U., 2006. A case for abductive reasoning over ontologies. In: OWLED
2. Shchekotykhin, K., Jannach, D., Schmitz, T., 2015. MergeXplain: Fast computation of multiple conflicts for diagnosis. In IJCAI
3. Homola, M., Pukancová, J., Gablíková, J., Fabianová, K., 2020. Merge, Explain, Iterate. In: DL

Vedúci:	Mgr. Júlia Pukancová, PhD.
Konzultant:	doc. RNDr. Martin Homola, PhD.
Katedra:	FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry:	prof. Ing. Igor Farkaš, Dr.
Dátum zadania:	11.10.2021

Dátum schválenia: 12.10.2021

prof. RNDr. Roman Ďurikovič, PhD.
garant študijného programu

.... študent

.... vedúci práce

I hereby declare that this master thesis is ...

. . . .

Bratislava, 2022 Bc. Janka Boborová

Acknowledgement

Text

Abstract

Text

Keywords: ...

Abstrakt

Text

Kľúčové slová: ...

Contents

Introduction

Introduction text.

Skúška citovania [16], [6], [4].

Part I

State of the art

Chapter 1

Description Logics

Description logics (DLs) [16, 2, 1] are a family of languages used for knowledge representation. They formally describe an application domain in a structured, clear and understandable way. The word *description* from the name is derived from the fact that an application domain is represented by concept *descriptors*, which are the expressions created from atomic concepts (unary predicates) and atomic roles (binary predicates) connected with the constructors of the specific DL language. The word *logic* from the name comes from the logic-based semantics of DLs that is pretty similar to first-order logic (FOL) semantics.

DLs have their predecessors, early knowledge representations, such as semantic networks and frames, which were very intuitive and comprehensive so they were readable and easily understood. On the other hand, they did not have a precise meaning and their interpretation could differ from person to person. This caused trouble while reasoning. Therefore, description logics were developed with the aim of preserving their intuitive representation and adding formal semantics to overcome the ambiguity of interpretation.

Unlike FOL, most DL languages are decidable (Def. 1.0.1). Research in the field of DL is mainly interested in decidable DL fragments, so decidability becomes a necessary condition for DL languages.

Definition 1.0.1 (Decidability [16]) *A class of problems is called decidable if there is a generic algorithm that can take any problem instance as an input and provide a yes-or-no answer after a finite time.*

In the context of logics, the common generic problem is entailment. If any of the logic problems is decidable, sometimes the logic itself is called decidable. As already mentioned, there are different DL languages. They differ in constructors that they use. More constructors typically mean more expressivity. In this chapter, we will focus on a specific DL language called $\mathcal{ALCHO}$, which is the most suitable for understanding our work. We will go through its syntax, semantics and other important notions.

1.1 Syntax of $\mathcal{ALCHO}$

DL is composed of three basic types of entities: individual names, concept names and role names. **Individual names** represent concrete objects from an application domain. For example, a concrete person tim, a pet fluffy or an object stool527. **Concept names** contain names that represent types or categories of objects that usually have similar properties. They also can be viewed as classes of objects. For instance, we can have concept name Person, Professor, Pet or Furniture. **Role names** contain names that represent binary relations that can occur between any two objects of a domain. For example, hasPet or teaches. We usually refer to these three sets of entities as **Vocabulary** (Def. 1.1.1).

Definition 1.1.1 (Vocabulary) *A DL vocabulary consists of three countable mutually disjoint sets:*

- *set of individuals* $N_I = \{a, b, ...\}$
- *set of atomic concepts* $N_C = \{A, B, ...\}$
- *set of roles* $N_R = \{R, S, ...\}$

According to a convention, concept names are capitalized while individual and role names are written in lowercase. Camel case is used for names that were created from multi-word notions.

Example 1.1.1 (Vocabulary) *Let us have a domain of pets and people. Vocabulary for this domain can be:*

- $N_I = \{\mathsf{tim}, \mathsf{eva}, \mathsf{erik}, \mathsf{fluffy}, \mathsf{pluto}, \mathsf{falco}\}$
- $N_C = \{\mathsf{Person}, \mathsf{Owner}, \mathsf{Good}, \mathsf{Happy}, \mathsf{Pet}, \mathsf{Dog}, \mathsf{Cat}\}$
- $N_R = \{\mathsf{likes}, \mathsf{owns}\}$

In $\mathcal{ALCHO}$ DL it is possible to define the concept by enumerating the individuals that should belong to it. These types of concepts are referred to as **nominals** (Def. 1.1.2).

Definition 1.1.2 (Nominals) *Nominals (nominal concepts) are concept expressions of the form:*

$$\{a_1, a_2, ..., a_n\}$$

where $\{a_1, a_2, ..., a_n\} \subseteq N_I$.

We can use nominals when we want to refer to some concrete individuals. That way we do not have to create an artificial concept name for these individuals.

Sometimes it is necessary to express more complex descriptions. We may, for instance, want to express happy people from our example domain without the need to create another concept name which would correspond to it. For these purposes, we have complex concepts. **Complex concepts** (Def. 1.1.3), also called non-atomic concepts or concepts, are recursively constructed from *atomic concepts* and *constructors.* Different DL languages use different sets of constructors to create complex concepts. Constructors used in $\mathcal{ALCHO}$ DL are: complement ($\neg$), intersection ($\sqcap$), union ($\sqcup$), existential restriction ($\exists$), value restriction ($\forall$).

The meaning of these constructors is very similar to the operations of the same name from set theory.

Definition 1.1.3 (Complex concepts) *Concepts are recursively constructed as the smallest set of expressions of the forms:*

$$C, D ::= A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists R.C \mid \forall R.C \mid \{a\}$$

where $a \in N_I, A \in N_C, R \in N_R$, and C, D are concepts.

Example 1.1.2 (Complex concepts and nominals) *We use the vocabulary from example 1.1.1.*

¬Person *This complex concept refers to all objects that are not people.*

Happy ⊓ Person *This complex concept refers to all objects that are happy and people at the same time. So, simply put, it refers to happy people.*

Person ⊔ Pet *This complex concept refers to all objects that are people or pets.*

$\exists$**hasPet**.**Dog** *This complex concept refers to all objects that are in relation* hasPet *with an object that is a dog. So, in natural language, we can say that it corresponds to all people that have a dog as a pet.*

$\forall$**likes**.{**falco**} *This complex concept refers to all objects for which holds that if they like anything it has to be the individual* falco*. So, it refers to all objects that like only* falco *or nothing at all.*

$\exists$**hasPet**.**Dog** $\sqcap$ $\forall$**likes**.{**falco**, **fluffy**} *We can also create a concept from complex concepts. This concept refers to all objects that have a dog as a pet and, at the same time, any object they like is either* falco *or* fluffy*.*

Another simplification, syntactic sugar, is the **top concept** and **bottom concept** (Def. 1.1.4). In some cases, we may want to refer to all objects from a domain (top concept), or, on the contrary, we want to express that something does not apply to any object, that is, to somehow express an empty set (bottom concept).

Definition 1.1.4 (Top and bottom concepts) *The top ($\top$) and bottom ($\bot$) concepts are defined as syntactic shorthands:*

- $\top$ *is a placeholder for* $A \sqcup \neg A$
- $\bot$ *is a placeholder for* $A \sqcap \neg A$

where A is any atomic concept.

Now that we are familiar with the basic entity types and concepts, it is time to look at how we formally represent domain knowledge. This domain knowledge is captured in a **knowledge base** (Def. 1.1.7).

We distinguish between two basic types of knowledge: *intensional knowledge* and *extensional knowledge*. Intensional knowledge is general knowledge or terminology of a domain that describes concepts and roles, and relationships between them. For example, we can have a relationship between the concepts Owner and Person: all owners are people, and that would be part of intensional knowledge. Extensional knowledge is knowledge about individuals and is often referred to as empirical knowledge or facts. For instance, we can say that individual fluffy is a pet.

Because of this knowledge division, the knowledge base is also divided into two parts: **TBox** (Def. 1.1.5) and **ABox** (Def. 1.1.6). TBox contains intensional knowledge while ABox contains extensional knowledge.

Definition 1.1.5 (TBox) *A TBox $\mathcal{T}$ is a finite set of GCI and RIA axioms ϕ of the form:*

- $\phi ::= C \sqsubseteq D$
- $\phi ::= R \sqsubseteq S$

where C, D are any concepts and $R, S \in N_R$.

The term GCI stands for general concept inclusion and term RIA for role inclusion axiom. They are subsumption axioms.

Definition 1.1.6 (ABox) *An ABox $\mathcal{A}$ is a finite set of assertion axioms (assertions) ϕ of the form:*

- $\phi ::= a\colon C$ *(concept assertion)*
- $\phi ::= a, b\colon R$ *(role assertion)*

where $a, b \in N_I, R \in N_R$, and C is any concept.

Let us also define assertion $a, b\colon \neg R$ as a shortcut for $a\colon \forall R.\neg b$, where $a, b \in N_I$ and $R \in N_R$. We will refer to this expression as a *negated role assertion.*

We can come across an alternative notation of assertions, that is often used. Concept assertion might be denoted as $C(a)$ where $a \in N_I$ and C is any concept. Role assertion might be denoted as $R(a, b)$ where $a, b \in N_I$ and $R \in N_R$. These alternative notations have the exact same meaning as those from the definition and might be used interchangeably.

Definition 1.1.7 (Knowledge base) *A DL knowledge base (KB) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is a pair consisting of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$.*

Example 1.1.3 (Knowledge base) *We will use vocabulary from example 1.1.1 and create a concise knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ about people and pets.*

$$\begin{aligned} \mathcal{T} = \{\mathsf{Owner} &\sqsubseteq \mathsf{Person}, \\ \mathsf{Owner} &\sqsubseteq \exists \mathsf{owns}.\mathsf{Pet}, \\ \mathsf{Dog} \sqcup \mathsf{Cat} &\sqsubseteq \mathsf{Pet}, \\ \mathsf{owns} &\sqsubseteq \mathsf{likes}, \\ \exists \mathsf{owns}.(\mathsf{Good} \sqcap \mathsf{Pet}) &\sqsubseteq \mathsf{Happy}\} \end{aligned}$$

$$\begin{aligned} \mathcal{A} = \{\mathsf{eva}\colon &\ \mathsf{Owner}, \\ \mathsf{fluffy}\colon &\ \mathsf{Dog}, \\ \mathsf{fluffy}\colon &\ \mathsf{Good}, \\ \mathsf{tim}, \mathsf{fluffy}\colon &\ \mathsf{owns}\} \end{aligned}$$

In some materials [16], we can also see a different division of the knowledge base: *Abox*, *TBox* and *RBox.* In such cases, the axioms that describe the roles and their relationships are contained in RBox. In $\mathcal{ALCHO}$ DL, RBox would contain RIA axioms. The TBox would then contain only descriptions of the concepts and their relationships, so GCI axioms. Thus, intensional knowledge would be divided into TBox and RBox.

If we were to apply this different division to our knowledge base from example 1.1.3, the only change would be to move the axiom $\mathsf{owns} \sqsubseteq \mathsf{likes}$ to RBox.

1.2 Semantics of $\mathcal{ALCHO}$

Now that we got familiar with the syntax, we will look at the $\mathcal{ALCHO}$ DL semantics. DLs semantics belongs to model-theoretic semantics. In this type of semantics, models are used to interpret symbols of the language. A model is a mathematical structure that represents some possible "state of the world". Usually, it is not the state of the whole world but only some small part, an application domain. A model maps symbols from the vocabulary to the model elements. It consists of a *domain* which contains elements and an *interpretation function* which ensures the mapping. We can imagine model elements to be some concrete objects of our world. They can be represented in different ways, e.g. numbers, words or pictures. The number of these elements can be infinite. The interpretation function then maps the symbols to elements from the model domain. Every symbol from the vocabulary has to have its interpretation. However, there is no need to use all elements from the domain to interpret something.

In DLs, these model structures are called **interpretations** (Def. 1.2.1) and they are used to describe the meaning of DL entities.

As we mentioned in the chapter introduction, the semantics of DLs are very similar to FOL semantics. FOL also has model-theoretic semantics. Instead of the term *interpretation*, FOL uses the term *structure* to denote model structure.

Definition 1.2.1 (Interpretation) *An interpretation of a given vocabulary is a pair* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *which contains:*

- *a non-empty* ***domain*** $\Delta^{\mathcal{I}}$
- *an* ***interpretation function*** $\cdot^{\mathcal{I}}$*, such that:*
  - $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ *for all* $a \in N_I$
  - $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ *for all* $A \in N_C$
  - $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ *for all* $R \in N_R$
- *for any concepts* C*,* D*, role* R *and individuals* $a_1, ..., a_n \in N_I$ *the* ***interpretation of complex concepts*** *is recursively defined as follows:*
  - $\neg C^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
  - $C \sqcap D^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $C \sqcup D^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $\exists R.C^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}} : (x, y) \in R^{\mathcal{I}} \wedge y^{\mathcal{I}} \in C^{\mathcal{I}}\}$
  - $\forall R.C^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y \in \Delta^{\mathcal{I}} : (x, y) \in R^{\mathcal{I}} \implies y^{\mathcal{I}} \in C^{\mathcal{I}}\}$
  - $\{a_1, ..., a_n\}^{\mathcal{I}} = \{a_1^{\mathcal{I}}, ..., a_n^{\mathcal{I}}\}$

Example 1.2.1 (Interpretation) *In the example, we will show an interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *of the vocabulary from example 1.1.1.*

$\Delta^{\mathcal{I}} = \{$ 🧒, 🧔, 👧, 👩, 👴, 🧓, 👵, 🐶, 🐕, 🐩, 🐕‍🦺, 🐱, 🐈, 🐈‍⬛ $\}$,

$$\begin{aligned}
\mathsf{tim}^{\mathcal{I}} &= \text{🧒}, & \mathsf{Person}^{\mathcal{I}} &= \{\text{👧}\}, & \mathsf{owns}^{\mathcal{I}} &= \{(\text{🧒},\text{🐶}),(\text{👧},\text{🐈‍⬛})\},\\
\mathsf{eva}^{\mathcal{I}} &= \text{👧}, & \mathsf{Owner}^{\mathcal{I}} &= \{\text{👧},\text{🧒}\}, & \mathsf{likes}^{\mathcal{I}} &= \{(\text{👧},\text{🐈‍⬛})\}\\
\mathsf{erik}^{\mathcal{I}} &= \text{👴}, & \mathsf{Happy}^{\mathcal{I}} &= \{\text{🧒}\}, & &\\
\mathsf{fluffy}^{\mathcal{I}} &= \text{🐶}, & \mathsf{Good}^{\mathcal{I}} &= \{\text{🐶}\}, & &\\
\mathsf{pluto}^{\mathcal{I}} &= \text{🐈‍⬛}, & \mathsf{Pet}^{\mathcal{I}} &= \{\text{🐈‍⬛}\}, & &\\
\mathsf{falco}^{\mathcal{I}} &= \text{🐕‍🦺}, & \mathsf{Dog}^{\mathcal{I}} &= \{\text{🐶}\}, & &\\
& & \mathsf{Cat}^{\mathcal{I}} &= \{\}, & &
\end{aligned}$$

Interpretation $\mathcal{I}$ is only one possible example. There are infinite possible interpretations of a given vocabulary, in which also different domains can be used.

In addition to symbols, we must also look at axioms in the context of interpretations. Axioms are some kind of statements usually given in the form of a knowledge base. Since the interpretation is some concrete state of the world, it is interesting for us to know whether these statements are valid in the given interpretation. So, we are interested in whether the axiom **is satisfied** in the given interpretation (Def. 1.2.2).

Definition 1.2.2 (Satisfaction) *Given an axiom ϕ, an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ satisfies ϕ ($\mathcal{I} \models \phi$) depending on its type:*

- $\mathbf{C} \sqsubseteq \mathbf{D}$: $\mathcal{I} \models C \sqsubseteq D$ *iff* $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathbf{R} \sqsubseteq \mathbf{S}$: $\mathcal{I} \models R \sqsubseteq S$ *iff* $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
- $\mathbf{a} \colon \mathbf{C}$: $\mathcal{I} \models a \colon C$ *iff* $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathbf{a}, \mathbf{b} \colon \mathbf{R}$: $\mathcal{I} \models a, b \colon R$ *iff* $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

where C, D are any concepts, R, $S \in N_R$ and a,b $\in N_I$.

Since we have defined the satisfaction of axioms, now we can determine whether a given interpretation satisfies a knowledge base. An interpretation that satisfies a knowledge base is called a **model** of a given knowledge base (not to be confused with the term *model as a mathematical structure* mentioned at the beginning of this section). We can imagine knowledge base models as the states of the world in which the statements from the knowledge base hold.

Definition 1.2.3 (Model) *An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a model of a DL KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ iff $\mathcal{I}$ satisfies every axiom in $\mathcal{T}$ and $\mathcal{A}$.*

Example 1.2.2 (Not a model) *Let us take interpretation $\mathcal{I}$ from example 1.2.1 and show that it is <u>not</u> a model of KB $\mathcal{K}$ from example 1.1.3. For $\mathcal{I}$ to be the model of the $\mathcal{K}$, it has to satisfy all axioms from $\mathcal{K}$. This is not the case. For example axiom* $\mathbf{Dog} \sqcup \mathbf{Cat} \sqsubseteq \mathbf{Pet}$ *is **<u>not</u> satisfied**:*

$$\mathcal{I} \models \mathsf{Dog} \sqcup \mathsf{Cat} \sqsubseteq \mathsf{Pet} \textit{ iff } \mathsf{Dog} \sqcup \mathsf{Cat}^{\mathcal{I}} \subseteq \mathsf{Pet}^{\mathcal{I}}$$

First, we formulate interpretations of the concepts $\mathsf{Dog} \sqcup \mathsf{Cat}$ *and* Pet.

$$\begin{aligned} \mathsf{Dog} \sqcup \mathsf{Cat}^{\mathcal{I}} &= \mathsf{Dog}^{\mathcal{I}} \cup \mathsf{Cat}^{\mathcal{I}} = \{\text{🐶}\}, \\ \mathsf{Pet}^{\mathcal{I}} &= \{\text{🐈‍⬛}\} \end{aligned}$$

We see that $\{\text{🐶}\} \nsubseteq \{\text{🐈‍⬛}\}$, so $\mathcal{I} \nvDash \mathsf{Dog} \sqcup \mathsf{Cat} \sqsubseteq \mathsf{Pet}$ and then $\mathcal{I} \nvDash \mathcal{K}$.

Note that some other axioms were also not satisfied, for example $\mathsf{owns} \sqsubseteq \mathsf{likes}$. *However, it is sufficient to show that it does not hold for at least one to prove that interpretation is not a model of $\mathcal{K}$.*

Example 1.2.3 (Model) *In this example we will show interpretation $\mathcal{I}_2 = (\Delta^{\mathcal{I}_2}, \cdot^{\mathcal{I}_2})$ and prove that it is a model of KB $\mathcal{K}$ from example 1.1.3.*

$$\Delta^{\mathcal{I}_2} = \{\ \text{🧒}, \text{🧔}, \text{👧}, \text{👩}, \text{👴}, \text{🧓}, \text{👵}, \text{🐶}, \text{🐕}, \text{🐩}, \text{🐕‍🦺}, \text{🐱}, \text{🐈}, \text{🐈‍⬛}\ \},$$

$$\begin{aligned}
\mathsf{tim}^{\mathcal{I}_2} &= \text{🧒}, & \mathsf{Person}^{\mathcal{I}_2} &= \{\text{👧},\text{🧒}\}, & \mathsf{owns}^{\mathcal{I}_2} &= \{(\text{🧒},\text{🐶}),(\text{👧},\text{🐈‍⬛})\},\\
\mathsf{eva}^{\mathcal{I}_2} &= \text{👧}, & \mathsf{Owner}^{\mathcal{I}_2} &= \{\text{👧},\text{🧒}\}, & \mathsf{likes}^{\mathcal{I}_2} &= \{(\text{🧒},\text{🐶}),(\text{👧},\text{🐈‍⬛})\},\\
\mathsf{erik}^{\mathcal{I}_2} &= \text{👴}, & \mathsf{Happy}^{\mathcal{I}_2} &= \{\text{🧒}\}, & &\\
\mathsf{fluffy}^{\mathcal{I}_2} &= \text{🐶}, & \mathsf{Good}^{\mathcal{I}_2} &= \{\text{🐶}\}, & &\\
\mathsf{pluto}^{\mathcal{I}_2} &= \text{🐈‍⬛}, & \mathsf{Pet}^{\mathcal{I}_2} &= \{\text{🐶},\text{🐈‍⬛}\}, & &\\
\mathsf{falco}^{\mathcal{I}_2} &= \text{🐕‍🦺}, & \mathsf{Dog}^{\mathcal{I}_2} &= \{\text{🐶}\}, & &\\
& & \mathsf{Cat}^{\mathcal{I}_2} &= \{\}, & &
\end{aligned}$$

In order to prove that $\mathcal{I}_2$ is indeed a model of $\mathcal{K}$, we must prove that it satisfies every axiom from $\mathcal{K}$. We start with assertions.

eva : Owner :

$\mathcal{I}_2 \models \mathsf{eva}\colon \mathsf{Owner}$ *iff* $\mathsf{eva}^{\mathcal{I}_2} \in \mathsf{Owner}^{\mathcal{I}_2}$

$\text{👧} \in \{\text{👧},\text{🧒}\}$, *so* $\mathcal{I}_2 \models \mathsf{eva}\colon \mathsf{Owner}$

fluffy : Dog :

$\mathcal{I}_2 \models \mathsf{fluffy}\colon \mathsf{Dog}$ *iff* $\mathsf{fluffy}^{\mathcal{I}_2} \in \mathsf{Dog}^{\mathcal{I}_2}$

$\text{🐶} \in \{\text{🐶}\}$, *so* $\mathcal{I}_2 \models \mathsf{fluffy}\colon \mathsf{Dog}$

fluffy : Good :

$\mathcal{I}_2 \models \mathsf{fluffy}\colon \mathsf{Good}$ *iff* $\mathsf{fluffy}^{\mathcal{I}_2} \in \mathsf{Good}^{\mathcal{I}_2}$

$\text{🐶} \in \{\text{🐶}\}$, *so* $\mathcal{I}_2 \models \mathsf{fluffy}\colon \mathsf{Good}$

tim, fluffy : owns :

$\mathcal{I}_2 \models \mathsf{tim}, \mathsf{fluffy}\colon \mathsf{owns}$ *iff* $(\mathsf{tim}^{\mathcal{I}_2}, \mathsf{fluffy}^{\mathcal{I}_2}) \in \mathsf{owns}^{\mathcal{I}_2}$

$(\text{🧒},\text{🐶}) \in \{(\text{🧒},\text{🐶}),(\text{👧},\text{🐈‍⬛})\}$, *so* $\mathcal{I}_2 \models \mathsf{tim}, \mathsf{fluffy}\colon \mathsf{owns}$

Now, we have to prove that subsumption axioms are also satisfied in $\mathcal{I}_2$.

Owner ⊑ Person :

$\mathcal{I}_2 \models \mathsf{Owner} \sqsubseteq \mathsf{Person}$ *iff* $\mathsf{Owner}^{\mathcal{I}_2} \subseteq \mathsf{Person}^{\mathcal{I}_2}$

$\{\text{👧},\text{🧒}\} \subseteq \{\text{👧},\text{🧒}\}$, *so* $\mathcal{I}_2 \models \mathsf{Owner} \sqsubseteq \mathsf{Person}$

Owner ⊑ ∃owns.Pet :

$\mathcal{I}_2 \models \mathsf{Owner} \sqsubseteq \exists\mathsf{owns.Pet}$ *iff* $\mathsf{Owner}^{\mathcal{I}_2} \subseteq \exists\mathsf{owns.Pet}^{\mathcal{I}_2}$

$\exists\mathsf{owns.Pet}^{\mathcal{I}_2} = \{x \in \Delta^{\mathcal{I}_2} \mid \exists y \in \Delta^{\mathcal{I}_2} : (x,y) \in \mathsf{owns}^{\mathcal{I}_2} \land y \in \mathsf{Pet}^{\mathcal{I}_2}\}$

$= \{x \in \Delta^{\mathcal{I}_2} \mid \exists y \in \Delta^{\mathcal{I}_2} : (x,y) \in \{(\text{🧒},\text{🐶}),(\text{👧},\text{🐈‍⬛})\} \land y \in \{\text{🐶},\text{🐈‍⬛}\}\}$

$= \{\text{👧},\text{🧒}\}$

$\{\text{👧},\text{🧒}\} \subseteq \{\text{👧},\text{🧒}\}$, *so* $\mathcal{I}_2 \models \mathsf{Owner} \sqsubseteq \exists\mathsf{owns.Pet}$

Dog ⊔ Cat ⊑ Pet :

$\mathcal{I}_2 \models \mathsf{Dog} \sqcup \mathsf{Cat} \sqsubseteq \mathsf{Pet}$ *iff* $\mathsf{Dog} \sqcup \mathsf{Cat}^{\mathcal{I}_2} \subseteq \mathsf{Pet}^{\mathcal{I}_2}$

$\mathsf{Dog} \sqcup \mathsf{Cat}^{\mathcal{I}_2} = \mathsf{Dog}^{\mathcal{I}_2} \cup \mathsf{Cat}^{\mathcal{I}_2} = \{\text{🐶}\}$

$\{\text{🐶}\} \subseteq \{\text{🐶},\text{🐈‍⬛}\}$, *so* $\mathcal{I}_2 \models \mathsf{Dog} \sqcup \mathsf{Cat} \sqsubseteq \mathsf{Pet}$

owns ⊑ likes :

$\mathcal{I}_2 \models \mathsf{owns} \sqsubseteq \mathsf{likes}$ *iff* $\mathsf{owns}^{\mathcal{I}_2} \subseteq \mathsf{likes}^{\mathcal{I}_2}$

$\{(\text{🧒},\text{🐶}),(\text{👧},\text{🐈‍⬛})\} \subseteq \{(\text{🧒},\text{🐶}),(\text{👧},\text{🐈‍⬛})\}$, *so* $\mathcal{I}_2 \models \mathsf{owns} \sqsubseteq \mathsf{likes}$

∃owns.(Good ⊓ Pet) ⊑ Happy :

$\mathcal{I}_2 \models \exists\mathsf{owns.}(\mathsf{Good} \sqcap \mathsf{Pet}) \sqsubseteq \mathsf{Happy}$ *iff* $\exists\mathsf{owns.}(\mathsf{Good} \sqcap \mathsf{Pet})^{\mathcal{I}_2} \subseteq \mathsf{Happy}^{\mathcal{I}_2}$

$\exists\mathsf{owns.}(\mathsf{Good} \sqcap \mathsf{Pet})^{\mathcal{I}_2} = \{x \in \Delta^{\mathcal{I}_2} \mid \exists y \in \Delta^{\mathcal{I}_2} : (x,y) \in \mathsf{owns}^{\mathcal{I}_2}$

$\land y \in \mathsf{Good} \sqcap \mathsf{Pet}^{\mathcal{I}_2}\}$

$= \{x \in \Delta^{\mathcal{I}_2} \mid \exists y \in \Delta^{\mathcal{I}_2} : (x,y) \in \mathsf{owns}^{\mathcal{I}_2} \land y \in \mathsf{Good}^{\mathcal{I}_2} \cap \mathsf{Pet}^{\mathcal{I}_2}\}$

$= \{x \in \Delta^{\mathcal{I}_2} \mid \exists y \in \Delta^{\mathcal{I}_2} : (x,y) \in \mathsf{owns}^{\mathcal{I}_2} \land y \in \{\text{🐶}\} \cap \{\text{🐶},\text{🐈‍⬛}\}\}$

$= \{x \in \Delta^{\mathcal{I}_2} \mid \exists y \in \Delta^{\mathcal{I}_2} : (x,y) \in \{(\text{🧒},\text{🐶}),(\text{👧},\text{🐈‍⬛})\}$

$\land y^{\mathcal{I}_2} \in \{\text{🐶}\}\}$

$= \{$🧒$\}$

$\{$🧒$\} \subseteq \{$🧒$\}$, *so* $\mathcal{I}_2 \models \exists \mathsf{owns}.(\mathsf{Good} \sqcap \mathsf{Pet}) \sqsubseteq \mathsf{Happy}$

We showed that $\mathcal{I}_2$ satisfies all axioms from $\mathcal{K}$, therefore we can say that $\mathcal{I}_2$ is a model of $\mathcal{K}$.

1.3 Basic decision problems

In the previous sections, we showed how to represent knowledge using language $\mathcal{ALCHO}$ DL. However, that is not the only thing we can do. Once we capture knowledge in a logical representation, a DL knowledge base, we can further reason with this knowledge.

The basic reasoning task is drawing *consequences* (implicit knowledge) from explicit knowledge that is captured in a knowledge base. This means that when we have a statement in the form of an axiom and we want to know whether this statement follows from a knowledge base. This decision problem is called **entailment** (Def.1.3.1).

Definition 1.3.1 (Entailment (Logical consequence)) *An axiom ϕ is entailed by a KB $\mathcal{K}$ (denoted $\mathcal{K} \models \phi$) iff for every $\mathcal{I}$, such that $\mathcal{I} \models \mathcal{K}$, holds $\mathcal{I} \models \phi$.*

In our case, it is enough to limit ourselves to problems related to ABox. Therefore, we will be interested in the entailment of assertion axioms. This decision problem is also referred to as **instance checking**. There are two types of assertion axioms, so we also have two types of instance checking: *concept instance checking* (Def. 1.3.2) and *role instance checking* (Def. 1.3.3).

Definition 1.3.2 (Concept instance checking) *An individual a is an instance of a concept C w.r.t. a DL KB $\mathcal{K}$ (denoted $\mathcal{K} \models a\colon C$) iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$ in all models $\mathcal{I}$ of $\mathcal{K}$.*

Definition 1.3.3 (Role instance checking) *A pair of individuals (a, b) is an instance of a role R w.r.t. a DL KB $\mathcal{K}$ (denoted $\mathcal{K} \models (a, b)\colon R$) iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$ in all models $\mathcal{I}$ of $\mathcal{K}$.*

Another important reasoning task is to decide if a knowledge base is consistent (Def. 1.3.4).

Definition 1.3.4 (ABox consistency) *A DL KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is consistent (also, $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$) iff it has at least one model.*

There may, indeed, be knowledge bases in which there is some kind of conflict and then it is not possible to construct an interpretation that would satisfy them. Let us show it in an example 1.3.1.

Example 1.3.1 (Inconsistent knowledge base) *We will use vocabulary from example 1.1.1. Let $\mathcal{K}_2 = (\mathcal{T}_2, \mathcal{A}_2)$ be our DL knowledge base, such that:*

$$\begin{aligned}\mathcal{T}_2 = \{&\mathsf{Dog} \sqcup \mathsf{Cat} \sqsubseteq \mathsf{Pet},\\ &\exists\mathsf{owns}.(\mathsf{Good} \sqcap \mathsf{Pet}) \sqsubseteq \mathsf{Happy}\}\end{aligned}$$

$$\begin{aligned}\mathcal{A}_2 = \{\mathsf{fluffy}&\colon \mathsf{Dog},\\ \mathsf{fluffy}&\colon \mathsf{Good},\\ \mathsf{tim}, \mathsf{fluffy}&\colon \mathsf{owns},\\ \mathsf{tim}&\colon \neg\mathsf{Happy}\}\end{aligned}$$

We can try to create a model of $\mathcal{K}_2$, but eventually, we will run into a conflict.

Individual fluffy *belongs to concept* Dog*, so from axiom* $\mathsf{Dog} \sqcup \mathsf{Cat} \sqsubseteq \mathsf{Pet}$ *it belongs to concept* Pet*. So individual* fluffy *belongs to concepts* Good *and* Pet*.*

Individual tim *is in relation* owns *with* fluffy*, therefore* tim *belongs to concept* $\exists\mathsf{owns}.(\mathsf{Good} \sqcap \mathsf{Pet})$*. Then, according to axiom* $\exists\mathsf{owns}.(\mathsf{Good} \sqcap \mathsf{Pet}) \sqsubseteq \mathsf{Happy}$*,* tim *must belong to concept* Happy*.*

In $\mathcal{K}_2$, however, tim *belongs to concept* $\neg\mathsf{Happy}$*. So, there is a conflict,* tim *cannot belong to* Happy *and* $\neg\mathsf{Happy}$ *at the same time. We are not able to create any model, so $\mathcal{K}_2$ is inconsistent.*

Some decision problems can be simplified and reformulated into another decision problem that is easier to solve. We can, for example, reduce the problem of instance checking to a consistency check (Lemmata 1.3.1, 1.3.2).

Lemma 1.3.1 (Reduction of concept instance checking) *Given a DL KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, an individual a and a concept C:*

$$\mathcal{K} \models a\colon C \textit{ iff } \mathcal{K}' = (\mathcal{T}, \mathcal{A} \cup \{a\colon \neg C\}) \textit{ is inconsistent.}$$

Lemma 1.3.2 (Reduction of role instance checking) *Given a DL KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, individuals a, b and a role R:*

$$\mathcal{K} \models (a,b)\colon R \textit{ iff } \mathcal{K}' = (\mathcal{T}, \mathcal{A} \cup \{(a,b)\colon \neg R\}) \textit{ is inconsistent.}$$

These lemmata come from an idea based on the definition of entailment and consistency. An entailed axiom must be satisfied in all models of knowledge base $\mathcal{K}$. If we add its negation to $\mathcal{K}$, then we should come to a conflict, because it is not possible to satisfy an axiom and its negation at the same time.

Example 1.3.2 (Concept instance checking) *We will use vocabulary from example 1.1.1. Let $\mathcal{K}_3 = (\mathcal{T}_3, \mathcal{A}_3)$ be our DL knowledge base, such that:*

$$\begin{aligned}\mathcal{T}_3 = \{&\mathsf{Dog} \sqcup \mathsf{Cat} \sqsubseteq \mathsf{Pet},\\ &\exists \mathsf{owns}.(\mathsf{Good} \sqcap \mathsf{Pet}) \sqsubseteq \mathsf{Happy}\}\end{aligned}$$

$$\begin{aligned}\mathcal{A}_3 = \{&\mathsf{fluffy} \colon \mathsf{Dog},\\ &\mathsf{fluffy} \colon \mathsf{Good},\\ &\mathsf{tim}, \mathsf{fluffy} \colon \mathsf{owns}\}\end{aligned}$$

Let us show that axiom **fluffy: Pet** ***is entailed by*** $\mathcal{K}_3$.

We will proceed according to lemma 1.3.1. Firstly, we construct a new knowledge base $\mathcal{K}'_3 = (\mathcal{T}_3, \mathcal{A}_3 \cup \{\mathsf{fluffy} \colon \neg\mathsf{Pet}\})$. Secondly, we show that $\mathcal{K}'_3$ is inconsistent.

We try to construct a model of $\mathcal{K}'_3$, but eventually, we will run into a conflict.

Individual fluffy *belongs to concept* Dog*, so from axiom* $\mathsf{Dog} \sqcup \mathsf{Cat} \sqsubseteq \mathsf{Pet}$ *he belongs to concept* Pet*. In $\mathcal{K}'_3$, however,* fluffy *belongs to concept* $\neg\mathsf{Pet}$*. So, there is a conflict because* fluffy *cannot belong to both* Pet *and* $\neg\mathsf{Pet}$*. We are not able to create any model, so $\mathcal{K}'_3$ is inconsistent and from that, $\mathcal{K}'_3 \models \mathsf{fluffy} \colon \mathsf{Pet}$.*

Finally, it is important to note that if a knowledge base is inconsistent, then it entails any axiom.

Chapter 2

Abduction

Text.

2.1 MHS algorithm

...

2.2 Our approach

...

2.2.1 MHS-MXP algorithm

...

2.2.2 QuickXplain

...

2.2.3 MergeXplain

...

Chapter 3

Original implementation

Text.

3.1 Problem of the original implementation

...

Chapter 4

Acquisition of a model

Text.

4.1 OWLKnowledgeExplorerReasoner

...

4.2 DIG Interface

...

4.3 JFact Reasoner

...

Part II

My contribution

Chapter 5

Implementation

Text.

Chapter 6

Evaluation

In this chapter, we will describe two experiments we conducted to test our MHS-MXP algorithm. In the first experiment, we will examine how the length and the number of explanations affect the algorithm runtime. In the second experiment, we will look at what results the algorithm achieves on a group of various real-world ontologies. In both experiments, we will compare our algorithm with the classic MHS algorithm.

In the previous work [3], various optimizations of the MHS-MXP algorithm were developed and tested. However, since none of them brought a significant improvement, we will use the classic version, which contains only Reiter's pruning optimizations.

We created one jar file of our implementation. Both algorithms, MHS and MHS-MXP, are included in our implementation. We can specify the algorithm in the input file. Since they are in the same implementation, their comparison is more accurate and there are no implementation differences.

However, we must note that this is not a plain version of the MHS algorithm, because it uses small optimizations (which are also used in MHS-MXP), such as removing unnecessary axioms from the models, which can reduce the size of the HS-tree that needs to be constructed.

6.1 Experiment 1

In the first experiment, we decided to repeat the part of the previous evaluation [3] where the focus is on comparing our MHS-MXP algorithm and the classic MHS algorithm. We also want to observe the effect of the explanations' length and their count on the MHS-MXP algorithm runtime.

We no longer have to modify the used ontology and create auxiliary concepts and axioms in it. This kind of approach was necessary for the previous version due to the limitations caused by the restrictive acquisition of models (described in chapter 4). Since this issue has been fixed, we can make a proper comparison with different types of observations in inputs.

Unlike the previous evaluation, we will also include inputs with negated assertions in the explanations. We have also created some new test cases that better suit our needs when examining the impact of the explanation count.

6.1.1 Dataset

In this experiment, we used the LUBM (Lehigh University Benchmark [9]) ontology for all our test cases. The application domain of this ontology is the university and it contains concepts such as Student, Institute, Employee, Publication and ResearchWork. The LUBM ontology is considered a standard benchmark for testing various reasoning capabilities of OWL knowledge base systems. The basic metrics of this ontology are shown in table 6.1.

concepts	**roles**	**individuals**	**GCI**	**RIA**	**logical axioms**
43	25	0	48	5	93

Table 6.1: Basic metrics of LUBM ontology.

6.1.2 Methodology

The LUBM ontology has a suitable size in terms of the concept count. However, it does not have a sufficiently complex structure, which is reflected in the length of explanations. As a result, simple observations in form of an atomic concept assertion return explanations with a length of 1 at most. In the experiment, we want to obtain explanations of different lengths. Therefore, we use complex observations, which were created as follows:

$$a\colon A_1 \sqcap ... \sqcap A_n$$

where $A_1, ..., A_n$ are atomic concepts from the ontology and a is a new individual (since the ontology does not contain any individuals).

An observation constructed in this way will have the largest explanation with a length of at most n. If $A_1, ..., A_n$ are independent and have no subconcepts, there will be no explanation (there would be only one possible explanation $\{A_1, ..., A_n\}$ which is not relevant). In case they are independent and have subconcepts, there will be at least 1 explanation and all obtained explanations will have length n. If these concepts are dependent, we get shorter explanations.

Test cases in the previous version of the evaluation [3] were generated so that $A_1, ..., A_n$ were always independent. They were grouped into 5 groups, S_1, S_2, S_3, S_4 and S_5, according to the length of the explanations, so that S_i contained all inputs with the explanations' length i. Each group contained 10 test cases. However, these groups were not evenly distributed as regards another parameter: the count of explanations. In this experiment, we would also like to emphasize the explanations count and its influence on the effectiveness of our algorithm, therefore we replaced some test cases with newly generated ones.

Evaluation process

...

6.1.3 Results

...

6.2 Experiment 2

In the second experiment, we wanted to test our algorithm on real-world ontologies, which would have different sizes and structure complexity. We were deeply inspired by article [12], which gave us suggestions on which ontologies we can choose from and how we can generate observations. We will also compare the MHS-MXP algorithm with the MHS algorithm again.

In this part, we will not allow explanations with negated assertions since, in Experiment 1, we showed their negative influence on our MHS-MXP algorithm. We already know that MHS-MXP has no advantage in such cases, so we focus on its strengths.

6.2.1 Dataset

The ontologies that we will use in the experiment were chosen from *ORE 2015 Reasoner Competition Corpus* [1]. This corpus was created to compare the capabilities of reasoners with various reasoning tasks.

Reasoner corpus contains 1920 ontology files. Among these ontologies, there are many which are not suitable for us. For example, there are in-

[1]https://zenodo.org/record/18578#.Y3tygXbMJPb

consistent ontologies in which there is no point in looking for explanations. Another example is ontologies for which the consistency check takes more than a minute (this is not suitable, because in our algorithm, the consistency check is called a considerable number of times). Therefore, we decided to explore these ontologies, find out their basic metrics and filter out the ones that are not suitable for us.

For the needs of this experiment, the sufficient number of ontologies is 20. Therefore, we could apply various filters to these ontologies in order to get only those that really interest us. First, we removed already mentioned inconsistent ontologies and ontologies whose consistency check took too long. Next, we filtered out excessively large ontologies that have more than 10,000 axioms. We removed ontologies that had less than 10 concept subsumptions. We also limited the number of individuals to a maximum of 100. The last limitation was the number of assertions to be at least 10 (they will be used for generating the observations).

In the end, we were left with 218 ontologies in the resulting group, and we randomly chose 20 of them. The basic metrics of selected ontologies are shown in table xx.

6.2.2 Methodology

In this experiment, we tried to generate observations non-randomly. ...

Evaluation process

...

6.2.3 Results

...

Conclusion

Conclusion text.

Bibliography

[1] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[2] F. Baader, I. Horrocks, C. Lutz, and U. Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.

[3] I. Balintová. Heuristic optimization of the mhs-mxp algorithm. Master's thesis, Comenius University in Bratislava, 2022.

[4] S. Bechhofer, R. Möller, and P. Crowther. The DIG description logic interface. In D. Calvanese, G. D. Giacomo, and E. Franconi, editors, *Proceedings of the 2003 International Workshop on Description Logics (DL2003), Rome, Italy September 5-7, 2003*, volume 81 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2003.

[5] I. Dickinson. Implementation experience with the dig 1.1 specification. *Hewlett Packard, Digital Media Sys. Labs, Bristol, Tech. Rep. HPL-2004-85*, 2004.

[6] C. Elsenbroich, O. Kutz, and U. Sattler. A case for abductive reasoning over ontologies. In B. C. Grau, P. Hitzler, C. Shankey, and E. Wallace, editors, *Proceedings of the OWLED*06 Workshop on OWL: Experiences

and Directions, Athens, Georgia, USA, November 10-11, 2006, volume 216 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.

[7] K. Fabianová. Optimization of an abductive reasoner for description logics. Master's thesis, Comenius University in Bratislava, 2019.

[8] J. Gablíková. Abductive reasoner for description logics combining mhs and mxp. Master's thesis, Comenius University in Bratislava, 2021.

[9] Y. Guo, Z. Pan, and J. Heflin. LUBM: A benchmark for OWL knowledge base systems. *J. Web Semant.*, 3(2-3):158–182, 2005.

[10] M. Homola, J. Pukancová, I. Balintová, and J. Boborová. Hybrid MHS-MXP abox abduction solver: First empirical results. In O. Arieli, M. Homola, J. C. Jung, and M. Mugnier, editors, *Proceedings of the 35th International Workshop on Description Logics (DL 2022) co-located with Federated Logic Conference (FLoC 2022), Haifa, Israel, August 7th to 10th, 2022*, volume 3263 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2022.

[11] M. Homola, J. Pukancová, J. Gablíková, and K. Fabianová. Merge, explain, iterate. In S. Borgwardt and T. Meyer, editors, *Proceedings of the 33rd International Workshop on Description Logics (DL 2020) co-located with the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020), Online Event [Rhodes, Greece], September 12th to 14th, 2020*, volume 2663 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.

[12] P. Koopmann, W. Del-Pinto, S. Tourret, and R. A. Schmidt. Signature-based abduction for expressive description logics. In D. Calvanese, E. Erdem, and M. Thielscher, editors, *Proceedings of the 17th International*

Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020, pages 592–602, 2020.

[13] J. Pukancová and M. Homola. Tableau-based abox abduction for the $\mathcal{ALCHO}$ description logic. In A. Artale, B. Glimm, and R. Kontchakov, editors, *Proceedings of the 30th International Workshop on Description Logics, Montpellier, France, July 18-21, 2017*, volume 1879 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2017.

[14] J. Pukancová and M. Homola. Abox abduction for description logics: The case of multiple observations. In M. Ortiz and T. Schneider, editors, *Proceedings of the 31st International Workshop on Description Logics co-located with 16th International Conference on Principles of Knowledge Representation and Reasoning (KR 2018), Tempe, Arizona, US, October 27th - to - 29th, 2018*, volume 2211 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018.

[15] R. Reiter. A theory of diagnosis from first principles. *Artif. Intell.*, 32(1):57–95, 1987.

[16] S. Rudolph. Foundations of description logics. In A. Polleres, C. d'Amato, M. Arenas, S. Handschuh, P. Kroner, S. Ossowski, and P. F. Patel-Schneider, editors, *Reasoning Web. Semantic Technologies for the Web of Data - 7th International Summer School 2011, Galway, Ireland, August 23-27, 2011, Tutorial Lectures*, volume 6848 of *Lecture Notes in Computer Science*, pages 76–136. Springer, 2011.

[17] K. M. Shchekotykhin, D. Jannach, and T. Schmitz. Mergexplain: Fast computation of multiple conflicts for diagnosis. In Q. Yang and M. J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International*

Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015, pages 3221–3228. AAAI Press, 2015.

List of Figures