Comenius university in bratislava

Faculty of Mathematics, Physics and Informatics

ABDUCTION SOLVER BASED ON HIGHLY EFFICIENT C++ DL REASONER

Master thesis

2018

Bc. Drahomír Mrózek

Comenius university in bratislava
Faculty of Mathematics, Physics and Informatics

ABDUCTION SOLVER BASED ON HIGHLY EFFICIENT C++ DL REASONER

Master thesis

Study programme:	Applied informatics
Field of study:	2511 Aplikovaná informatika
Study department:	Department of Applied Informatics
Advisor:	RNDr. Martin Homola, PhD.
Consultant:	Mgr. Júlia Pukancová

Bratislava, 2017

Bc. Drahomír Mrózek

todo:zadanie

Čestne prehlasujem, že túto diplomovú prácu som vypracoval samostatne len s použitím uvedenej literatúry a za pomoci konzultácií u môjho školiteľa.

. . . .

Bratislava, 2017 Bc. Drahomír Mrózek

Acknowledgments

TODO

Abstract

This work will start from an existing solution of an abduction solver based on Pellet reasoner for description logics, implemented in Java. The implementation in C++ opens different ways for improvement and effectivization, starting with the utilization of a more effective C++ inference system for description logics.
Keywords: abduction, description logic

Abstrakt

Práca bude vychádzať z existujúceho riešenia abduktívneho systému založeného na reasoneri pre deskripčné logiky Pellet implementovaného v Jave. Pri implementácii v C++ sa otvárajú možnosti na zlepšenie a zefektívnenie existujúceho návrhu abduktívneho solvera, počnúc využitím efektívnejšieho C++ inferenčného systému pre deskripčné logiky.

Kľúčové slová: abdukcia, deskripčná logika

Contents

1 Introduction 1

2 Description logic 2

2.1 Introduction to description logic 2

2.2 Selected description logics 5

2.2.1 $\mathcal{ALC}$ 5

2.2.2 $\mathcal{SHIQ}$ 8

2.2.3 $\mathcal{ALCHO}$ 10

2.2.4 $\mathcal{SROIQV}(\mathcal{D})$ 11

2.3 tableau algorithm 11

2.4 OWL (mozno) 12

2.5 Použitia (možno, spomenúť rôzne databázy) 12

3 Abduction 13

3.1 Definition 13

3.2 Uses 13

4 Previous abduction solvers 14

4.1 theoretical approaches 14

4.2 implemented solutions 14

5 Our approach **15**

5.1 Explanation of our algorithm 15

6 Implementation **16**

7 Results **19**

8 Conclusion **20**

Chapter 1

Introduction

Story No. 1 TODO Úvod, troska kontextu, asi na 1,5 strany Cieľom práce je návrh a vývoj abduktívneho systému pre deskripčné logiky založený na existujúcom reasoneri s dôrazom na optimalizačné techniky a efektívnosť.

Chapter 2

Description logic

2.1 Introduction to description logic

This chapter is based on Handbook on ontologies [5] and Handbook of knowledge representation [6].

The word “ontology” is used with different meanings in different communities. In philosophy, Aristotle in his Metaphysics defined **Ontology** as the study of attributes that belong to things because of their very nature.
Ontology focuses on the nature and structure of things independently of any further considerations, and even independently of their actual existence.
For example, it makes perfect sense to study the Ontology of unicorns and other fictitious entities: although they do not have actual existence, their nature and structure can be described in terms of general categories and relations.

in Computer Science, we refer to an **ontology** as a special kind of information object or computational artifact. Computational ontologies are a means to formally model the structure of a system, that is the relevant entities and relations that emerge from its observation, and which are useful to our purposes. An example of such a system can be a company with all its employees and their interrelationships. The ontology engineer analyzes relevant entities and organizes them into concepts and relations, being represented, respectively, by unary and binary predicates.

Description logics (**DLs**) are a family of knowledge representation languages that can be used to represent an ontology in a structured and formally well-understood way. The "description" part of their name is based on how the important notions of the domain are described by concept descriptions (unary predicates) and atomic roles (binary predicates). The "logic" part comes from their formal, logic-based semantics, unlike some other methods of representation of ontologies, for example semantic networks.
Knowledge base (a set of facts) in description logics typically comes in two parts: a terminological part (**TBox**) and an assertional part(**ABox**).
TBox consists of general statements about concepts. Some examples,:

Example 2.1.1 *[6]* $HappyMan \equiv Human \sqcap \neg Female \sqcap (\exists married.Doctor) \sqcap (\forall hasChild.(Doctor \sqcup Professor))$.

This example defines a concept, 'HappyMan', as a human who is not female, is married to a doctor and his every child is a doctor or a professor.

Example 2.1.2 *[6]* $\exists hasChild.Human \sqsubseteq Human$

Or, in natural language, if someone has a child that is human, then they are human.
ABox consists of specific statements about individuals.

Example 2.1.3 *[6] bob : HappyMan*

bob,mary : hasChild

mary : $\neg$*Doctor*

This is an **ABox** of 3 statements: Bob is a happy man, Bob has a child - Mary, and Mary is not a doctor. You may notice that if we had a knowledge base consisting of TBox 2.1.1 and ABox 2.1.3, we may deduce that Mary must be a professor.

Interpretation of description logics is done using sets. We will formally define interpretations with specific description logics, but to informally make sense of previous examples:

Concepts can be interpreted as sets of constants,

individuals can be interpreted as constants,

relations as a set of pairs of constants,

$\sqcap$ as set conjunction $\cap$,

$\sqcup$ as set disjunction $\cup$,

$\neg$ as set complement,

$\sqsubseteq$ as subset symbol $\subseteq$,

existential restriction $\exists$**r.C** as a set of constants that are in relation **r** with at least one individual in concept **C**,

and **universal restriction** $\forall$ **r.C** as a set of constants that are not in relation **r** with any constant in complement of **C**.

Also, A $\equiv$ B means "A $\sqsubseteq$ B and B $\sqsubseteq$ A".

2.2 Selected description logics

2.2.1 $\mathcal{ALC}$

In this thesis, we will be using a widely used description logic $\mathcal{ALC}$ and it's extensions. $\mathcal{ALC}$ stands for "Attributive concept Language with Complements". It's one of the less expressive languages, for example, it can't express the concept " someone who has 2 children ". You can see examples of statements in $\mathcal{ALC}$ in the previous section.

Definition 2.2.1 *[6] (Syntax of $\mathcal{ALC}$ concepts and roles). Let N_C be a set of concept names and N_R be a set of role names. The set of Concepts is the smallest set such that*

1. $\top, \bot$, *and every concept name $A \in N_C$ is an Concept,*
2. *If C and D are Concepts and $r \in N_R$, then $C \sqcap D$, $C \sqcup D, \neg C, \forall r.C$, and $\exists r.C$ are Concepts.*

$\top$ and $\bot$ are special concepts 'everything' and 'nothing'. Every individual belongs to concept $\top$ and no individuals belong to concept $\bot$.

The semantics of $\mathcal{ALC}$ (and of DLs in general) are given in as interpretations.

Definition 2.2.2 *[6] ($\mathcal{ALC}$ semantics). An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a nonempty set $\Delta^{\mathcal{I}}$, called the domain of $\mathcal{I}$, and a function $\cdot^{\mathcal{I}}$ that maps every $\mathcal{ALC}$ Concept to a subset of $\Delta^{\mathcal{I}}$, and every $\mathcal{ALC}$ role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ such that, for all $\mathcal{ALC}$ Concepts C, D and all role names r:*

$$
\begin{aligned}
&\top^{\mathcal{I}} = \Delta^{\mathcal{I}} \quad \bot^{\mathcal{I}} = \emptyset, \\
&(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}, \\
&(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
&\neg C^{\mathcal{I}} = \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}, \\
&(\exists r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} | \exists y \in \Delta^{\mathcal{I}} \textit{ with } \langle x, y\rangle \in r^{\mathcal{I}} \textit{ and } y \in C^{\mathcal{I}}\}, \\
&(\forall r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} | \forall y \in \Delta^{\mathcal{I}}, \textit{ if } \langle x, y\rangle \in r^{\mathcal{I}}, \textit{ then } y \in C^{\mathcal{I}}\}.
\end{aligned}
$$

You may notice that nothing in the definition of interpretation says that an ontology must be 'true'. An interpretation which can intuitively be called 'true' for an ontology is called a model. We will now formally define it. An ontology in description logic is often called 'knowledge base'. It consists of various statements, in $\mathcal{ALC}$ it consists of general concept inclusions (GCI), and assertional axioms. A set of GCIs are usually called a TBox (example **??**) and a set of assertional axioms ABox(example 2.1.3).

Definition 2.2.3 *[6] ($\mathcal{ALC}$ TBox model)*
A general concept inclusion (GCI) axiom is of the form $C \sqsubseteq D$, where C, D are $\mathcal{ALC}$ Concepts. An interpretation $\mathcal{I}$ is a model of a GCI $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.
$\mathcal{I}$ is a model of a TBox T if it is a model of every GCI in T.

Definition 2.2.4 *[6] ($\mathcal{ALC}$ ABox model)*
An assertional axiom is of the form $x : C$ or $(x, y) : r$, where C is an $\mathcal{ALC}$ Concept, r is a role name, and x and y are individual names. An interpretation $\mathcal{I}$ is a model of an assertional axiom $x : C$ if $x^{\mathcal{I}} \in C^{\mathcal{I}}$, and $\mathcal{I}$ is a model of an assertional axiom $(x, y) : r$ if $(x^{\mathcal{I}}, y^{\mathcal{I}}) \in r^{\mathcal{I}}$.
$\mathcal{I}$ is a model of an ABox A if it is a model of every assertional axiom in A .

Definition 2.2.5 *[6] (Consistency)*
$\mathcal{I}$ is a model of a knowledge base $\mathcal{K}$=(Tbox $\mathcal{T}$, Abox $\mathcal{A}$) if it's a model of $\mathcal{A}$ and $\mathcal{T}$.
If a model of $\mathcal{K}$ exists, we say that $\mathcal{K}$ is consistent.

An ontology can have multiple models, some less intuitive than other.

Example 2.2.1 *(model)*

Knowledge base $\mathcal{K} = \{\mathcal{T}, \mathcal{A}\}$, $\mathcal{T}$= {

$A \sqsubseteq B$

$A \sqsubseteq \exists r.C$

$C \sqsubseteq \forall r.D$

}

$\mathcal{A}$ = {

a : A

c,d : r

}

One possible model of $\mathcal{K}$, $\mathcal{M}_1$, *would be:* {

$\Delta^{\mathcal{I}}$ = { a_x, c_x, d_x }

$a^{\mathcal{I}} = a_x, c^{\mathcal{I}} = c_x, d^{\mathcal{I}} = d_x$

$A^{\mathcal{I}} = \{a_x\}, B^{\mathcal{I}} = \{a_x\}, C^{\mathcal{I}} = \{c_x\}, D^{\mathcal{I}} = \{d_x\}$

$r^{\mathcal{I}} = \{\langle a_x, b_x\rangle, \langle a_x, c_X\rangle, \langle c_x, d_x\rangle\}$

}

But other models also exist, for example $\mathcal{M}_2$ *and* $\mathcal{M}_3$. $\mathcal{M}_2$ = {

$\Delta^{\mathcal{I}}$ = { a_x, c_x, d_x, c_n }

$a^{\mathcal{I}}$ = $a_x, c^{\mathcal{I}} = c_x, d^{\mathcal{I}} = d_x$

$A^{\mathcal{I}}$ = $\{a_x\}, B^{\mathcal{I}} = \{a_x\}, C^{\mathcal{I}} = \{c_n\}, D^{\mathcal{I}} = \{\}$

$r^{\mathcal{I}} = \{\langle a_x, b_x\rangle, \langle a_x, c_n\rangle, \langle c_x, d_x\rangle\}$ }

$\mathcal{M}_3$ = {

$\Delta^{\mathcal{I}} = \{i_x\}$

$a^{\mathcal{I}} = i_x, c^{\mathcal{I}} = i_x, d^{\mathcal{I}} = i_x$

$A^{\mathcal{I}} = B^{\mathcal{I}} = C^{\mathcal{I}} = D^{\mathcal{I}} = \{i_x\}$

$r^{\mathcal{I}} = \{\langle i_x, i_x\rangle\}$

}

2.2.2 $\mathcal{SHIQ}$

$\mathcal{SHIQ}$ is one of the most expressive description logics. **S** - abbreviation of $\mathcal{ALC}$ with transitive roles.

H - Role hierarchy (role r_1 can be subrole of role r_2)

I - Inverse properties (if a,b : r, then b,a : r^-)

Q - Quantified cardinality restrictions (for example $\leq 2hasChild$)

Examples of Concepts in $\mathcal{SHIQ}$:

Example 2.2.2 *[5]* $Human \sqcap \neg Female \sqcap \exists married.Doctor$
$\sqcap (\geq 5hasChild) \sqcap \forall hasChild.Professor$.

"A man that is married to a doctor and has at least five children, all of whom are professors".

Example 2.2.3 *[5]* $Human \sqsubseteq \forall hasParent.Human \sqcap (\geq 2hasParent.\top)$
$\sqcap (\leq 2hasParent.\top) \sqcap \forall hasParent^-.Human$

"If someone is a human, all their parents are human. they have exactly two parents, and everything that has them as a parent (i.e. is their child) is a human."

Example 2.2.4 *[5]*
$hasParent \sqsubseteq hasAnccestor.$

"hasParent is a subrole of hasAncestor (i. e. If A hasparent.B , then A hasAncestor.B)."

Example 2.2.5 *Trans(hasAncestor)*

"The role hasAncestor is transitive (i.e. if A hasAncestor.B and B hasAncestor.C then A hasAncestor.C)."

The definitions of syntax and semantics of $\mathcal{SHIQ}$ are similar to those of $\mathcal{ALC}$.

Definition 2.2.6 *[5] ($\mathcal{SHIQ}$ concept and role syntax) Let R be a set of role names, which is partitioned into a set R_+ of transitive roles and a set R_p of normal roles. The set of all $\mathcal{SHIQ}$ roles is $R \cup \{r^- | r \in R\}$, where r^- is called the inverse of the role r.*
Let C be a set of concept names. The set of $\mathcal{SHIQ}$ concepts is the smallest set such that:

1. *every concept $A \in C$ is a $\mathcal{SHIQ}$ concept.*

2. *if A and B are $\mathcal{SHIQ}$ concepts and r is a $\mathcal{SHIQ}$ role, then $A \sqcap B, A \sqcup B, \neg A, \forall r.A, and \exists r.A$ are $\mathcal{SHIQ}$ concepts.*

3. *if A is a $\mathcal{SHIQ}$ concept and r is a simple $\mathcal{SHIQ}$ role (simple role is neither transitive nor has a transitive subrole), and $n \in \mathbb{N}$, then $(\leq nr.A)$ and $(\geq nr.A)$ are $\mathcal{SHIQ}$ concepts.*

$\mathcal{SHIQ}$ semantics can be described as $\mathcal{ALC}$ semantics with same additions.

Definition 2.2.7 *($\mathcal{SHIQ}$ semantics) [5] in addition to definition* **??**, *for all* $p \in R$ *and* $r \in R_+$:
$\langle x, y\rangle \in p^{\mathcal{I}}$ *iff* $\langle y, x\rangle \in (p^-)^{\mathcal{I}}$.
if $\langle x, y\rangle \in r^{\mathcal{I}}$ *and* $\langle y, z\rangle \in r^{\mathcal{I}}$ *then* $\langle x, z\rangle \in r^{\mathcal{I}}$.
$(\leq nr.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} | \#r^{\mathcal{I}}(x, C) \leq n\}$,
$(\geq nr.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} | \#r^{\mathcal{I}}(x, C) \geq n\}$,

where $\#M$ *denotes the cardinality of the set M, and* $r^{\mathcal{I}}(x, C) := \{y | \langle x, y\rangle \in r^{\mathcal{I}}$ *and* $y \in C^{\mathcal{I}}\}$.

$\mathcal{SHIQ}$ ABox and it's model are the same as in $\mathcal{ALC}$ (definition 2.2.4). For the TBox, we have to add role subsumption axioms.

Definition 2.2.8 *[5] ($\mathcal{SHIQ}$ TBox)*
A role inclusion axiom is of the form $r \sqsubseteq s$, *where r,s are roles. An interpretation* $\mathcal{I}$ *is a model of a role inclusion axiom* $r \sqsubseteq s$ *if* $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$.
$\mathcal{I}$ *is a model of a TBox* $\mathcal{T}$ *if it it is a model of every role inclusion and GCI axioms (definition* **??**) *in TBox.*

2.2.3 $\mathcal{ALCHO}$

$\mathcal{ALC}$ with role hierarchy and Nominals.
Nominals are concepts with exacly one specific instance. For example, {john} is a concept with its only instance being the individual 'john'.
Nominals can be used to express enumerations, for example [2]:
$Beatle \equiv \{john\} \sqcup \{paul\} \sqcup \{george\} \sqcup \{ringo\}$

Role hirerarchy was already described in section 2.2.2 (example 2.2.4).

Definition 2.2.9 *(Syntax and semantics of $\mathcal{ALCHO}$)*
Syntax of $\mathcal{ALCHO}$ is the syntax of $\mathcal{ALC}$ (definition 2.2.1), with {a} added to the set of concepts C for each individual 'a'.
Similarly, semantics of $\mathcal{ALCHO}$ are the semantics of $\mathcal{ALC}$ (definition 2.2.2) with the addition of $\{a\}^{\mathcal{I}} = \{a^{\mathcal{I}}\}$,
where 'a' is an individual and $\mathcal{I}$ is the interpretation.

In $\mathcal{ALCHO}$, the definition of an ABox is the same for $\mathcal{ALC}$ (definition 2.2.4) and the definition of a TBox is the same as for $\mathcal{SHIQ}$ (definition 2.2.8.

2.2.4 $\mathcal{SROIQV}(\mathcal{D})$

Jazyk koncludu, TODO asi ked bude hotova implementacia.

2.3 tableau algorithm

In our algorithm, we heavily make use of tableau algorithm. Tableau algorithm is a method of constructing a model of a knowledge base $\mathcal{K}$ if $\mathcal{K}$ is consistent, and stops if $\mathcal{K}$ is inconsistent and thererfore no model exists.

Tableau algorithm uses knowledge base in negation normal form (NNF), that is, every concept complement $\neg$ applies only to a concept name [6]. Any $\mathcal{ALC}$ concept can be transformed to an equivalent concept in NNF by using de Morgan's laws and the duality between existential and universal restrictions ($\neg\exists r.C \equiv \forall r.\neg C$). For example, the concept $\neg(\exists r.A \sqcap \forall s.B)$, where A, B are concept names, can be transformed using de Morgan's laws to $\neg\exists r.C \sqcup \neg\forall s.B$, and this can then be transformed using the existential-universal duality into $(\forall r.\neg A) \sqcup (\exists s.\neg B)$.

The idea behind the tableau algorithm for $\mathcal{K} = \{\mathcal{T}, \mathcal{A}\}$is to start with the concrete situation described in $\mathcal{A}$ and expand based on what can be inferred from $\mathcal{T}$ and currently known ABox statements. This is done by starting with a tree where nodes are individuals, directed edges are relations between individuals, and each individual has a label consisting of concepts the individual belongs to and rules that apply to the individual. The algorithm must at some times, specifically when $\sqcup$ rule is used, make an undeterministic decision.

Clash happens when, for an a concept C, both C and $\neg C$ are in an individuals label. if this happens (in a deterministic implementation), we backtrack to a previous undeterministic decision with an unexplored choice and continue from there. If there is no decision with an unexplored possibility we can backtrack to, we can declare $\mathcal{K}$ to be inconsistent. On the other hand, if all relevant rules have been used for each individual, then $\mathcal{K}$ is consistent as we have just found it's model.
TODO: priklad s obrazkom, formalna definicia, casova zlozitost, rozsirenie tableau pre ostatne spomenute deskripcne logiky.

2.4 OWL (mozno)

TODO: definovat SHOIN

2.5 Použitia (možno, spomenúť rôzne databázy)

Chapter 3

Abduction

3.1 Definition

3.2 Uses

Chapter 4

Previous abduction solvers

4.1 theoretical approaches

‘

4.2 implemented solutions

Chapter 5

Our approach

5.1 Explanation of our algorithm

Chapter 6

Implementation

```
function ABDUCTION( Knowledge base $\mathcal{K}$, observation O, maximum Depth maxD)
    Output: Set of minimum explanations $\mathcal{S}$
    if $\mathcal{K} \cup O$ is inconsistent then
        return $\emptyset$ //observation not consistent with knowledge base
    end if
    if $\mathcal{K} \cup \neg O$ is inconsistent then
        return $\{\{\}\}$ //observation can be inferred from knowledge base
    end if
    $\mathcal{C} = \{\{\}\}$ //hitting set candidates for this iteration (one empty set)
    $\mathcal{NC} = \emptyset$ //hitting set candidates for next iteration
    $\mathcal{S} = \emptyset$
    D = 1 //depth
    while $\mathcal{C} \neq \emptyset \wedge D \leq maxD$ do
        for each candidate $c \in \mathcal{C}$ //TODO: malo by c tiez byt v mathcal? do
            for each hitting set $s \in \mathcal{S}$ do
                if $s \in c$ then
                    Continue to next candidate //expalanation wouldn't be minimal
                end if
            end for
            if $\mathcal{K} \cup \{\neg O\} \cup c$ is inconsistent then
                if $\mathcal{K} \cup c$ is consistent then
                    $\mathcal{S} = \mathcal{S} \cup c$ //c is a hitting set (explanation)
                end if
                Continue to next candidate
            end if
            if D = maxD then
                Continue to next candidate // no need to create candidates for next while iteration
            end if
            Axioms $\mathcal{AX}$ = getAntiModel($\mathcal{K}, O, c$)
            for each axiom $ax \in \mathcal{AX}$ do
                if $ax \in c$ then
                    Continue to next axiom //$c \cup ax \equiv c$
                end if
                $nc = c \cup ax$
                if $O \in nc$ then
                    Continue to next axiom //explanation would be trivial
                end if
                $\mathcal{NC} = \mathcal{NC} \cup nc$
            end for
        end for
        $\mathcal{C} = \mathcal{NC}$
        $\mathcal{NC} = \emptyset$
    end while
end function
```

function GETANTIMODEL(Knowledge base $\mathcal{K}$, observation O, set of axioms $\mathcal{AX}$)

  Output: Antimodel $\mathcal{AM}$ of a model $\mathcal{M}$ of $\mathcal{K} \cup \{\neg O\} \cup \mathcal{AX}$

  $\mathcal{AM} = \emptyset$

  $\mathcal{M}$ = model of $\mathcal{K} \cup \{\neg O\} \cup \mathcal{AX}$

  for each individual $\mathcal{I} \in \mathcal{K} \cup O \cup \mathcal{AX}$: **do**

    $\mathcal{C}_k$ =set of concepts $\{\forall C|\mathcal{I} : C \in \mathcal{K}\ \}$ //known concepts

    $\mathcal{C}_a$ = set of all concepts $\in \mathcal{K}$ //all concepts

    $\mathcal{C}_i$ = set of concepts $\{\forall C|\mathcal{I} : C \in \mathcal{M}\ \}$ //inferred concepts

    for each concept $C \in C_i$ **do**

      if not $C \in \mathcal{C}_k$ **then**

        $\mathcal{AM} = \mathcal{AM} \cup (\mathcal{I} : C)$

      end if

    end for

    for each concept $C \in \mathcal{C}_a$ **do**

      if not $C \in \mathcal{C}_i$ **then**

        $\mathcal{AM} = \mathcal{AM} \cup (\mathcal{I} : C)$

      end if

    end for

  end for

  return $\mathcal{AM}$

end function

Chapter 7

Results

Chapter 8

Conclusion

Literatúra

[1] K. Halland and K. Britz. Naive abox abduction in alc using a dl tableau. In Y. Kazakov, D. Lembo, and F. Wolter, editors, *Description Logics*, volume 846 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[2] M. Krötzsch, F. Simancik, and I. Horrocks. A description logic primer. *arXiv preprint arXiv:1201.4089v3*, 2012.

[3] J. Pukancová and M. Homola. Tableau-based abox abduction for description logics, 2017.

[4] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[5] S. Staab and R. E. Studer. *Handbook on Ontologies*. Springer, 2010.

[6] H. Van, V. L. Frank, and P. e. Bruce. *Kandbook of knowledge representation*. Elsevier, 2008.

Zoznam obrázkov