ABox Abduction for Description Logics: The Case of Multiple Observations

Júlia Pukancová and Martin Homola

Comenius University in Bratislava
Faculty of Mathematics, Physics, and Informatics,
Mlynská dolina, 84248 Bratislava
{pukancova,homola}@fmph.uniba.sk

Abstract. We develop an ABox abduction algorithm for description logics based on Reiter's minimal hitting set algorithm. It handles abduction problems with multiple observations and it supports the class of explanations allowing atomic and negated atomic concept and role assertions. As shorter explanations are preferred, the algorithm computes shorter explanations first and allows to limit their length. The algorithm is sound and complete for this class of explanations and for any given maximal length of explanations. In this paper we specifically focus on computing explanations for abduction problems with multiple observations, for which we explore two distinct approaches. The DL expressivity is limited only by the DL reasoner that our algorithm calls as a black box. We provide an implementation on top of Pellet, which is a full OWL 2 reasoner, so the expressivity is up to $\mathcal{SROIQ}$. We evaluate the implementation on three different ontologies.

Keywords: description logics · abduction · implementation · evaluation

1 Introduction

The goal of abduction [20,5] is to explain why a set of axioms $\mathcal{O}$ (called observation) does not follow from a knowledge base $\mathcal{K}$: an explanation for $\mathcal{O}$ is another set of axioms $\mathcal{E}$ s.t. $\mathcal{O}$ follows from $\mathcal{K} \cup \mathcal{E}$. ABox abduction, i.e. the case when both $\mathcal{O}$ and $\mathcal{E}$ are limited to ABox axioms, has found applications in areas such as diagnostic reasoning [16,21,5] or multimedia interpretation [6,2].

On the other hand, general-purpose ABox abduction solvers, especially for more expressive DLs are still underdeveloped. Some approaches are based on translation, e.g., Klarman et al. [18] and Du et al. [4] rely on a translation to first-order and modal logic, respectively to logic programs. The former work is purely theoretical, it is sound and complete, but limited to $\mathcal{ALC}$. The latter has some interesting computational results but it is only complete w.r.t. a specific Horn fragment of $\mathcal{SHIQ}$. Other works, e.g., of Halland and Britz [12,11] and Ma et al. [19] directly exploit DL tableau reasoning, but their expressivity is still limited to $\mathcal{ALC}$ and $\mathcal{ALCI}$. The former work is a theoretical proposal, it is sound, but not complete. The latter was implemented, but soundness or completeness is not shown. An ABox abduction service based on backward chaining of DL-safe rules [6,2] is part of RacerPro [10]. To our best knowledge, soundness and

completeness results are not known. An interesting proposal based on forgetting was recently developed by Del-Pinto and Schmidt [3]. Their work includes an implementation, it is sound and complete for $\mathcal{ALC}$, and it handles semantic, not just syntactic minimality.

Building on ideas of Halland and Britz [11,12], we have developed an ABox abduction solver AAA [22] based on Reiter's [23] Minimal Hitting Set (MHS) algorithm. AAA supports atomic and negated atomic concept and role assertions in explanations. It uses a DL reasoner as a black box, hence it handles any DL expressivity supported by the reasoner – Pellet in case of our implementation. It also exploits optimization techniques suggested by Reiter such as model reuse and pruning.

In this paper, we extend our previous results as follows: (a) we develop two distinct approaches for computing explanations for abduction problems with multiple explanations; (b) we incorporate a limitation on maximum length of explanations and improve effective exploration of the search space, starting from more desired (i.e., shorter) explanations; (c) we conduct preliminary empirical evaluation, specifically focusing on the comparison of the two multiple-observation approaches and on evaluation of the implemented optimization techniques.

2 Preliminaries

For brevity we will only introduce the $\mathcal{ALCHO}$ DL [1] which contains all features essential to our approach – especially due to constructions involved in handling multiple observations and role explanations. The lowest expressivity that the DL reasoner used in our abduction algorithm should support is $\mathcal{ALCO}$. Role hierarchies are not strictly needed, but without them the number of explanations involving roles is limited.

A DL vocabulary consists of three countable mutually disjoint sets: set of individuals $N_\mathrm{I} = \{a, b, \ldots\}$, set of atomic concepts $N_\mathrm{C} = \{A, B, \ldots\}$, and set of roles $N_\mathrm{R} = \{R, S, \ldots\}$. Complex $\mathcal{ALCHO}$ *concepts* are recursively constructed as stated in Table 1, where C, D are $\mathcal{ALCHO}$ concepts, R, S are roles, and a, b are individuals. A TBox $\mathcal{T}$ is a finite set of GCIs and RIAs, and an ABox $\mathcal{A}$ is a finite set of concept, role or negated role assertions, as given in Table 1. A knowledge base is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. By the negation of any assertion φ, i.e. $\neg\varphi$, we mean $\neg C(a)$ for $\varphi = C(a)$, $\neg R(a, b)$ for $\varphi = R(a, b)$, and $R(a, b)$ for $\varphi = \neg R(a, b)$. Let $\neg\mathcal{A} = \{\neg\varphi \mid \varphi \in \mathcal{A}\}$ for any ABox $\mathcal{A}$.

An *interpretation* of a knowledge base $\mathcal{K}$ is a pair $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I} \neq \{\}$, and $\cdot^\mathcal{I}$ is an interpretation function s.t. $a^\mathcal{I} \in \Delta^\mathcal{I}$ for $a \in N_\mathrm{I}$, $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ for $A \in N_\mathrm{C}$, and $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$ for $R \in N_\mathrm{R}$. Interpretation of complex concepts is inductively defined in Table 1. $\mathcal{I}$ *satisfies* an axiom φ ($\mathcal{I} \vDash \varphi$) as given in Table 1. $\mathcal{I}$ is a *model* of $\mathcal{K}$ if it satisfies all axioms included in $\mathcal{K}$; $\mathcal{K}$ is *consistent* if there is a model $\mathcal{I}$ of $\mathcal{K}$. A concept C is *satisfiable* w.r.t. $\mathcal{K}$ if there is a model $\mathcal{I}$ of $\mathcal{K}$ s.t. $C^\mathcal{I} \neq \{\}$. An axiom φ is *entailed* by $\mathcal{K}$ ($\mathcal{K} \vDash \varphi$) if for every model $\mathcal{I}$ of $\mathcal{K}$ it holds that $\mathcal{I} \vDash \varphi$. A set of axioms $\mathcal{O}$ is entailed by $\mathcal{K}$ ($\mathcal{K} \vDash \mathcal{O}$) if $\mathcal{K} \vDash \varphi$ for all $\varphi \in \mathcal{O}$.

Entailment and consistency checking are inter-reducible: e.g., for an ABox assertion φ, $\mathcal{K} \vDash \varphi$ if and only if $\mathcal{K} \cup \{\neg\varphi\}$ is inconsistent. There is a number of DL reasoners [14,15,25,24,7,26] for consistency checking, using the tableau algorithm (TA) [1].

In DL we distinguish between TBox and ABox abduction [5], the latter is of our interest in this paper. An *ABox abduction problem* is a pair $\mathcal{P} = (\mathcal{K}, \mathcal{O})$, where $\mathcal{K}$ is

Table 1. Syntax and Semantics of $\mathcal{ALCHO}$

Constructor	Syntax	Semantics
complement	$\neg C$	$\Delta^I \setminus C^I$
intersection	$C \sqcap D$	$C^I \cap D^I$
union	$C \sqcup D$	$C^I \cup D^I$
existential restriction	$\exists R.C$	$\{x \mid \exists y\,(x,y) \in R^I \wedge y \in C^I\}$
value restriction	$\forall R.C$	$\{x \mid \forall y\,(x,y) \in R^I \rightarrow y \in C^I\}$
nominal	$\{a\}$	$\{a^I\}$
Axiom	Syntax	Semantics
concept incl. (GCI)	$C \sqsubseteq D$	$C^I \subseteq D^I$
role incl. (RIA)	$R \sqsubseteq S$	$R^I \subseteq S^I$
concept assertion	$C(a)$	$a^I \in C^I$
role assertion	$R(a,b)$	$(a^I, b^I) \in R^I$
neg. role assertion	$\neg R(a,b)$	$(a^I, b^I) \notin R^I$

a DL knowledge base and $\mathcal{O}$ is a set of ABox assertions. A set of ABox assertions $\mathcal{E}$ is an explanation of $\mathcal{P}$ if $\mathcal{K} \cup \mathcal{E} \models \mathcal{O}$. If $\mathcal{O} = \{O\}$ we call $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ a *single-observation* abduction problem, and we also denote it by $\mathcal{P} = (\mathcal{K}, O)$. Otherwise $\mathcal{P}$ is called a *multiple-observation* abduction problem.

Not all explanations are acceptable [5]. We will require an explanation $\mathcal{E}$ of $\mathcal{P}$ to be *explanatory* ($\mathcal{K} \not\models \mathcal{O}$), *consistent* ($\mathcal{E} \cup \mathcal{K}$ is consistent), and *relevant* (for each $O_i \in \mathcal{O}$: $\mathcal{E} \not\models O_i$). Even after ruling out such undesired explanations, there can still be too many of them. Therefore some notion of minimality is often used. We will use syntactic minimality: an explanation $\mathcal{E}$ of $\mathcal{P}$ is *minimal* if there is no other explanation $\mathcal{E}'$ of $\mathcal{P}$ s.t. $\mathcal{E}' \subsetneq \mathcal{E}$.

In this work we are interested in explanations in form of atomic and negated atomic ABox assertions. More specifically, we require $\mathcal{E} \subseteq \{A(a), \neg A(a), R(a,b), \neg R(a,b) \mid A \in N_C, R \in N_R, a, b \in N_I\}$. This class of explanations will be denoted $\mathrm{A_nR_n}$. The subset of this class, which only contains explanatory, consistent, relevant, and minimal explanations will be denoted $\mathrm{A_nR_n^{CER,sub}}$.

Note that for any abduction problem $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ both $\mathcal{K}$ and $\mathcal{O}$ are finite and the class of explanations is defined w.r.t. their finite combined signature, hence there is only finitely many explanations in either of these classes for any abduction problem $\mathcal{P}$.

3 ABox Abduction with Single Observation

To compute explanations for single-observation abduction problems we build on our previous work [22] which in turn extends the works of Halland and Britz [12,11] and the seminal work of Reiter [23].

In order to find an explanation for an ABox abduction problem $\mathcal{P} = (\mathcal{K}, O)$ we need to find a set of ABox assertions $\mathcal{E}$ such that $\mathcal{K} \cup \mathcal{E} \models O$, i.e., such that $\mathcal{K} \cup \mathcal{E} \cup \{\neg O\}$ is inconsistent. This can be done by finding a *hitting set* [17] of the collection $\mathcal{M}$ of all models of $\mathcal{K} \cup \{\neg O\}$ and then negating each assertion in the hitting set.

To stay within the class $\mathrm{A_nR_n}$ we will rely on *ABox encoding* of models. ABox encoding of $\mathcal{I}$ is $M_\mathcal{I} = \{C(a) \mid \mathcal{I} \models C(a), C \in \{A, \neg A\}, A \in N_\mathrm{C}, a \in N_\mathrm{I}\} \cup \{R(a,b) \mid \mathcal{I} \models R(a,b),\ R \in N_\mathrm{R},\ a,b \in N_\mathrm{I}\} \cup \{\neg R(a,b) \mid \mathcal{I} \models \neg R(a,b),\ R \in N_\mathrm{R},\ a,b \in N_\mathrm{I}\}$. Similarly as above, each $\mathcal{K}$ and O have a finite combined signature, hence each ABox encoding is finite. What is more, $M_\mathcal{I}$ is not necessary homomorphic with the original model $\mathcal{I}$; it ignores the anonymous part of the model (on purpose). Hereafter we automatically assume the ABox encoding whenever we talk about models.

According to Reiter [23], hitting sets are found by constructing a HS-tree $T = (V, E, L)$, a labelled tree which contains negated models of $\mathcal{K} \cup \{\neg O\}$ as node-labels and whose edges are labelled by ABox assertions contained in their parent node. HS-tree has the property that the node label $L(n)$ and the union of the edge-labels $H(n)$ on the path from root r to each node n is disjoint. If n cannot be extended by any successor satisfying this property then $H(n)$ corresponds to a hitting set.

Additional pruning conditions apply on the HS-tree [23,22]. The most obvious case is when a node n already corresponds to a hitting set $H(n)$ and there is another node n' with $H(n) \subseteq H(n')$. Any such n' can be pruned. For further conditions refer to literature [23,22]. A pruned HS-tree (i.e., one on which all pruning conditions were applied), once completed, contains all minimal hitting sets [23].

Theorem 1. *Given an ABox abduction problem $\mathcal{P} = (\mathcal{K}, O)$. Let $\mathcal{M}$ be a set of all models of $\mathcal{K} \cup \{\neg O\}$ and let $T = (V, E, L)$ be a pruned HS-tree for a collection of sets $\{\neg M \mid M \in \mathcal{M}\}$. Then $\{H(n) \mid n \in V$ is a leaf in T that is not pruned$\}$ is the collection of all minimal explanations of $\mathcal{P}$.*

The Single Observation Abduction algorithm (SOA) is given in Algorithm 1. This is a simplified and shortened pseudocode with some details omitted. For a more detailed version please refer to our previous report [22].

The algorithm takes a knowledge base $\mathcal{K}$ and an observation O as first two inputs. One improvement w.r.t. the previous work [22] is that the search can be limited only to explanations up to certain length, this is done by the third parameter l. The current version also implements model reuse [23,22] and it has a few additional improvements, notably it adds additional pruning, but the details are beyond the scope of this paper.

The last two parameters are auxiliary and they are only relevant when the algorithm is called as a subroutine to find explanations for multiple-observation abduction problems. In the single-observation case they will be set to $\mathcal{O} = \{O\}$ and some new individual s_0 (w.r.t. both $\mathcal{K}$ and O).

The algorithm first tries to obtain a model of $\mathcal{K} \cup \{\neg O\}$ to use $\neg M$ as a label for the root node r. If there is none (i.e., $\mathcal{K} \models O$) no (explanatory) explanations exist. Otherwise, successors of r are initialized and the tree is traversed breadth-first down to the depth l. For each node, we first evaluate $H(n)$. If it is inconsistent, irrelevant, or pruning applies, the node is pruned.

Next, if $\mathcal{K} \cup \{\neg O\} \cup H(n)$ has a model, it is used to label n, otherwise we have found an explanation which is stored. Finally, after the desired depth l is fully traversed, all explanations are returned.

The algorithm calls the external solver (TA) on lines 1, 10, 11 and 15 though some TA calls can be saved thanks to model reuse. Also, while pruning ensures minimality, we have also filtered out all undesired explanations, hence the following result.

Algorithm 1 SOA($\mathcal{K}$,O,l,$\mathcal{O}$,s_0)

Require: knowledge base $\mathcal{K}$, observation O, max explanation length l, set of observations $\mathcal{O}$, individual s_0
Ensure: set $\mathcal{S}_\mathcal{E}$ of all explanations of $\mathcal{P} = (\mathcal{K}, O)$ of the class $\mathrm{A_n R_n^{CER,sub}}$ up to the length l

$M \leftarrow$ a model M of $\mathcal{K} \cup \{\neg O\}$ obtained from TA
if $M =$ `null` **then**
    return `"nothing to explain"`
end if
create new HS-tree $T = (V, E, L)$ with root r
label r by $L(r) \leftarrow \neg M \setminus \{\varphi \in M \mid s_0 \text{ occurs in } \varphi\}$
for each $\sigma \in \neg M$ create a successor n_σ of r and label the resp. edge by σ
$\mathcal{S}_\mathcal{E} \leftarrow \{\}$
while there is next node n in T **and** $|H(n)| \leq l$ **do**
    if $\mathcal{K} \cup H(n)$ is inconsistent
        or $H(n) \cup \{\neg O_i\}$ is inconsistent for some $O_i \in \mathcal{O}$
        or n can be pruned
    then
        prune n
    else if there is a model M of $\mathcal{K} \cup \{\neg O\} \cup H(n)$, reused or obtained from TA **then**
        label n by $L(n) \leftarrow \neg M \setminus \{\varphi \in M \mid s_0 \text{ occurs in } \varphi\}$
    else
        $\mathcal{S}_\mathcal{E} \leftarrow \mathcal{S}_\mathcal{E} \cup \{H(n)\}$
    end if
    for each $\sigma \in \neg M$ create a successor n_σ of n and label the resp. edge by σ
end while
return $\mathcal{S}_\mathcal{E}$

Theorem 2. *Let $\mathcal{P} = (\mathcal{K}, O)$ be a single-observation ABox abduction problem, $l \geq 1$, and s_0 a new individual w.r.t. $\mathcal{K}$ and O. SOA($\mathcal{K}, O, l, \{O\}, s_0$) terminates, and it is sound and complete w.r.t. all explanations of $\mathcal{P}$ of the class $\mathrm{A_n R_n^{CER,sub}}$ up to the length l.*

In addition, if we remove the length limitation (i.e., we set l to ∞), the algorithm still terminates as the depth of the HS-tree is also bound by the number of possible ABox assertions. In such a case the algorithm finds all explanations of the input ABox abduction problem.

4 Handling Multiple Observations

We now extend SOA to handle multiple observations. This algorithm is called ABox Abduction Algorithm (AAA). In fact, we explore two versions, one based on reducing the set of observations to a single observation ($\mathrm{AAA_R}$), the other based on splitting the multiple-observation problem into separate single-observation subproblems ($\mathrm{AAA_S}$).

4.1 Reduction

An observation consisting of multiple concept assertions involving the same individual, say $\mathcal{O} = \{C_1(a), \ldots, C_n(a)\}$, can be easily reduced to an equivalent single observation

$O' = C_1 \sqcap \cdots \sqcap C_n(a)$. Cases involving multiple individuals or even role assertions are less straightforward, but in DLs featuring nominals they may be encoded as follows.

Lemma 1. *Let $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ be a multiple-observation ABox abduction problem with $\mathcal{O} = \{C_1(a_1), \ldots, C_n(a_n), R_1(b_1, c_1), \ldots, R_m(b_m, c_m), \neg Q_1(d_1, e_1), \ldots, \neg Q_l(d_l, e_l)\}$, where C_i are concepts, $R_i, Q_j \in N_\mathrm{R}$ and $a_i, b_j, c_j, d_k, e_k \in N_\mathrm{I}$ for $i \in [1, n]$, $j \in [1, m]$, $k \in [1, l]$. Let $\mathcal{E} \subseteq \{A(a), \neg A(a), R(a, b), \neg R(a, b) \mid A \in N_\mathrm{C}, R \in N_\mathrm{R}, a, b \in N_\mathrm{I}\}$. Let $\mathcal{P}' = (\mathcal{K}, O')$ be a single-observation ABox abduction problem with $O' = X(s_0)$, s.t. s_0 is new w.r.t. $\mathcal{K}$, $\mathcal{O}$, and $\mathcal{E}$, and*

$$\begin{aligned} X = &(\neg\{a_1\} \sqcup C_1) \sqcap \cdots \sqcap (\neg\{a_n\} \sqcup C_n) \\ &\sqcap (\neg\{b_1\} \sqcup \exists R_1.\{c_1\}) \sqcap \cdots \sqcap (\neg\{b_m\} \sqcup \exists R_m.\{c_m\}) \\ &\sqcap (\neg\{d_1\} \sqcup \forall Q_1.\neg\{e_1\}) \sqcap \cdots \sqcap (\neg\{d_l\} \sqcup \forall Q_l.\neg\{e_l\}) \,. \end{aligned}$$

Then $\mathcal{E}$ is an explanation of $\mathcal{P}$ if and only if it is an explanation of $\mathcal{P}'$.

Notice that the lemma rules out explanations involving the individual s_0 introduced during the reduction. If this is not the case $\mathcal{P}'$ may indeed have more explanations than $\mathcal{P}$. These unwanted explanations need to be filtered out.

Example 1. Let $\mathcal{K} = \{A \sqsubseteq B, C \sqsubseteq D\}$, and $\mathcal{O} = \{B(a), D(b)\}$. The only explanation of $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ is $\mathcal{E}_1 = \{A(a), C(b)\}$. Using the reduction we obtain $\mathcal{P}' = (\mathcal{K}, O')$ with $O' = (\neg\{a\} \sqcup B) \sqcap (\neg\{b\} \sqcup D)(s_0)$. However, besides for $\mathcal{E}_1$ which is an explanation of $\mathcal{P}'$ courtesy of Lemma 1, in addition $\mathcal{E}_2 = \{A(s_0), C(s_0)\}$, $\mathcal{E}_3 = \{A(a), C(s_0)\}$, and $\mathcal{E}_4 = \{A(s_0), C(b)\}$ are explanations of $\mathcal{P}'$.

The $\mathrm{AAA_R}$ algorithm is listed in Algorithm 2. It takes a knowledge base $\mathcal{K}$, a set of observations $\mathcal{O}$, and a length upper bound $l \geq 1$ as inputs. It reduces the set $\mathcal{O}$ to a single observation O' according to Lemma 1 and passes the reduced single-observation abduction problem to SOA.

Algorithm 2 $\mathrm{AAA_R}$ ($\mathcal{K}$,$\mathcal{O}$,l): AAA based on Reduction

Require: knowledge base $\mathcal{K}$, set of observations $\mathcal{O} = \{C_1(a_1), \ldots, C_n(a_n), R_1(b_1, c_1), \ldots, R_m(b_m, c_m), \neg Q_1(d_1, e_1), \ldots, \neg Q_l(d_l, e_l)\}$, max length of an explanation l
Ensure: set $\mathcal{S}_\mathcal{E}$ of all explanations of $\mathcal{P} = (\mathcal{K}, O)$ of the class $\mathrm{A_n R_n^{CER,sub}}$ up to the length l
1: $s_0 \leftarrow$ new individual w.r.t. $\mathcal{K}$ and $\mathcal{O}$
2: $X \leftarrow (\neg\{a_1\} \sqcup C_1) \sqcap \cdots \sqcap (\neg\{a_n\} \sqcup C_n)$
$\sqcap (\neg\{b_1\} \sqcup \exists R_1.\{c_1\}) \sqcap \cdots \sqcap (\neg\{b_m\} \sqcup \exists R_m.\{c_m\})$
$\sqcap (\neg\{d_1\} \sqcup \forall Q_1.\neg\{e_1\}) \sqcap \cdots \sqcap (\neg\{d_l\} \sqcup \forall Q_l.\neg\{e_l\})$
3: $O' \leftarrow X(s_0)$
4: $\mathcal{S}_\mathcal{E} \leftarrow \mathrm{SOA}(\mathcal{K}, O', l, \mathcal{O}, s_0)$
5: **return** $\mathcal{S}_\mathcal{E}$

Instead of filtering the unwanted explanation involving the auxiliary individual s_0 ex post, $\mathrm{AAA_R}$ passes s_0 to SOA as the fifth parameter. SOA then excludes the assertions involving s_0 already from the models returned by TA, reducing the HS-tree.

Since for multiple-observation abduction the relevance needs to be checked w.r.t. all observations $\mathrm{AAA_R}$ also passes the original set of observations $\mathcal{O}$ to SOA as the fourth parameter. Observing that SOA handles the auxiliary parameters correctly, the correctness of $\mathrm{AAA_R}$ is then a consequence of the correctness of SOA.

Theorem 3. *Let $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ be a multiple-observation ABox abduction problem, and let $l \geq 1$. Then AAA$_R$ ($\mathcal{K}$,$\mathcal{O}$,l) terminates, and it is sound and complete w.r.t. all explanations of $\mathcal{P}$ of the class $\mathrm{A_nR_n^{CER,sub}}$ up to the length l.*

As for SOA, if we remove the depth limitation, the algorithm finds all explanations of the input ABox abduction problem.

4.2 Splitting into Subproblems

A multiple-observation abduction problem may also be solved by splitting $\mathcal{P}$ into n subproblems $\mathcal{P}_i = (\mathcal{K}_i, O_i)$ and answering each $\mathcal{P}_i$ separately using SOA. If there is no bound on length, this is quite easy: we simply combine the results in terms of union with some additional filtering [22]. But the partial explanations may overlap or even repeat. If a length bound l is given, we need to run SOA up to l for each subproblem, and only then combine the results. Only this assures all explanations up to the length l for $\mathcal{P}$ (plus possibly some which are longer). This may seem as unnecessary overhead compared to $\mathrm{AAA_R}$ but as we show below, sometimes it may be useful.

$\mathrm{AAA_S}$ algorithm is listed in Algorithm 3. It receives a knowledge base $\mathcal{K}$, observations $\mathcal{O} = \{O_1, \ldots, O_n\}$, and a length upper bound $l \geq 1$ as inputs.

Algorithm 3 $\mathrm{AAA_S}$ ($\mathcal{K}$,$\mathcal{O}$,l): AAA with Splitting

Require: knowledge base $\mathcal{K}$, set of observations $\mathcal{O}$, max explanation length l
Ensure: set $\mathcal{S}_\mathcal{E}$ of all explanations of $\mathcal{P} = (\mathcal{K}, O)$ of the class $\mathrm{A_nR_n^{CER,sub}}$ up to the length l

$s_0 \leftarrow$ new individual w.r.t. $\mathcal{K}$ and $\mathcal{O}$ ▷ auxiliary
$\Sigma \leftarrow \{\}$ ▷ collection of partial results
$\mathcal{K}' \leftarrow \mathcal{K} \cup \{\mathsf{T}(a) \mid a \text{ occurs in } O_i, O_i \in \mathcal{O}\}$
for all $O_i \in \mathcal{O}$ **do**
  $\mathcal{S}_{\mathcal{E}_i} \leftarrow \mathrm{SOA}(\mathcal{K}', O_i, l, \mathcal{O}, s_0)$ ▷ get partial result for O_i
  if $\mathcal{S}_{\mathcal{E}_i} = \{\}$ **then**
    return $\{\}$ ▷ $\mathcal{O}$ has no explanation
  else if $\mathcal{S}_{\mathcal{E}_i} \neq$ `"nothing to explain"` **then**
    $\Sigma \leftarrow \Sigma \cup \{\mathcal{S}_{\mathcal{E}_i}\}$ ▷ store partial result
  end if
end for
if $\Sigma = \{\}$ **then**
  return `"nothing to explain"`
else
  $\mathcal{S}_\mathcal{E} \leftarrow \{\mathcal{E}_1 \cup \cdots \cup \mathcal{E}_m \mid \mathcal{E}_i \in \mathcal{S}_{\mathcal{E}_i}, \mathcal{S}_{\mathcal{E}_i} \in \Sigma, m = |\Sigma|\}$
  $\mathcal{S}_\mathcal{E} \leftarrow \{\mathcal{E} \in \mathcal{S}_\mathcal{E} \mid \mathcal{E} \text{ is minimal, consistent, and relevant}\}$
end if
return $\mathcal{S}_\mathcal{E}$

The algorithm initializes an empty collection Σ which will be used to accumulate the partial results returned by SOA and a dummy new individual s_0 (lines 1–2).

We cannot just directly use $\mathcal{K}$ in each subproblem $\mathcal{P}_i$, as we may miss explanations involving individuals from other observations. Hence $\mathcal{K}'$ is used, obtained by adding $\top(a)$ into $\mathcal{K}$ for all such individuals a (line 3).

The algorithm then loops through $O_i \in \mathcal{O}$ (lines 4–11) and calls SOA for $\mathcal{K}'$, O_i and l. We also pass the set of observations $\mathcal{O}$ (due to relevance checks) and a dummy new individual s_0 to SOA as auxiliary parameters. If one of the observations cannot be explained (SOA returned $\{\}$) then neither the whole set $\mathcal{O}$ can be explained (line 7).

Observe that $\mathcal{O}$ and $\mathcal{O} \setminus \{O_i \in \mathcal{O} \mid \mathcal{K} \models O_i\}$ have the same set of explanations. Therefore if SOA returned `"nothing to explain"` for some O_i the result is simply excluded from Σ. If this happens for all $O_i \in \mathcal{O}$, the overall result is `"nothing to explain"`.

If Σ is non-empty the explanations of $\mathcal{P}$ are computed as unions $\mathcal{E}_1 \cup \cdots \cup \mathcal{E}_m$, combining all possible $\mathcal{E}_i$ from each $\mathcal{S}_{\mathcal{E}_i} \in \Sigma$ with the others (line 15). While SOA already did some filtering of the partial results, supersets, irrelevance, and inconsistence may have been introduced by unifying them, hence they are filtered out (line 16).

Finally, also $\mathrm{AAA_S}$ is correct up to any length l. And if the length limitation is removed, the algorithm finds all explanations of the input ABox abduction problem.

Theorem 4. *Let $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ be a multiple-observation ABox abduction problem, and let $l \geq 1$. Then $\mathrm{AAA_S}$ ($\mathcal{K}$,$\mathcal{O}$,l) terminates, and it is sound and complete w.r.t. all explanations of $\mathcal{P}$ of the class $\mathrm{A_n R_n^{CER,sub}}$ up to the length l.*

5 Evaluation

We have implemented the ABox abduction algorithm into the AAA solver. For implementation details, see our previous report [22]. The implementation is available at: `http://dai.fmph.uniba.sk/~pukancova/aaa/`. We present evaluation results with the aim to compare the two versions of the multiple-observation algorithm.

5.1 Dataset and Methodology

We have chosen three ontologies for the evaluation: Family ontology (our own small ontology of family relations)[1], Coffee ontology by Carlos Mendes[2], and LUBM (Lehigh University Benchmark [9]). The parameters of the ontologies are stated in Table 2.

On each input we ran AAA iteratively, rising maximal explanation length (i.e. the maximal depth of the HS-tree). The following properties were recorded from each run: time of execution, number of explanations, number of the nodes in the HS-tree, number of TA calls, number of reused models, number of pruned nodes.

Apart from exceptional cases all experiments were repeated 10 times on the same input and the results were averaged. All experiments were executed both allowing and disallowing loops (i.e., reflexive role assertions) in explanations. For the lack of space

[1] `http://dai.fmph.uniba.sk/~pukancova/aaa/ont/`

[2] `https://gist.githubcom/cmendesce/56e1e16aee5a556a186f512eda8dabf3`

Table 2. Parameters of the ontologies

Ontology	Concepts	Roles	Individuals	Axioms
Family ontology	8	1	2	24
Coffee ontology	41	6	2	291
LUBM	43	25	1	46

we report only the former in this paper. On average, the experiments with loops took 243.24 % more time and found 26.99 % more explanations.

All experiments were done on a 6-core 3.2 GHz AMD Phenom™ II X6 1090T Processor, 8 GB RAM, running Ubuntu 17.10, Linux 4.13.0, while the maximum Java heap size was set to 4GB. We have used the GNU `time` utility to measure the CPU time consumed by AAA while running in user mode, summed over all threads.

5.2 Experiment

The algorithm was executed once on each ontology with a specifically chosen observation set, each consisting of three assertions: {Father(jack), Mother(eva), Person(fred)} for Family ontology, {Milk(a), Coffee(b), Pure(c)} for Coffee ontology, {Person(jack), Employee(jack), Publication(a)} for LUBM. The aim was to compare the reduction (R) and the splitting (S) approach therefore all experiments were run with either option.

The experiment was processed up to the depth 3, as even in this depth the algorithm ran out of memory in half of the cases. The reason for this is that the observations contain multiple individuals which increases the search space. Most runs were repeated 10 times, with four exceptions: LUBM (R and S), depth 3 and Coffee (R), depth 3 as these runs ran out of memory; and Coffee (S) as the execution time was too high.

In Figure 1 the execution times for each ontology and both approaches are plotted. These times are also stated in Table 3 with the numbers of the computed explanations in each experiment. The out-of-memory cases show the time when the memory was exceeded. The average deviation in time between runs was 2.66 % for the splitting approach and 1.6 % for the reduction approach.

Table 3. Parameters of HS-trees

		Family			Coffee			LUBM		
App.	**D**	**Nodes**	**Expl.**	**Time**	**Nodes**	**Expl.**	**Time**	**Nodes**	**Expl.**	**Time**
S	1	93.0	7	4.54	480.0	12	23.46	411.0	320	32.08
	2	1545.0	144	21.78	41303.0	12	826.74	39756.3	320	421.21
	3	17072.6	813	1169.02	2052996.0	12	332177.08	–	–	–
R	1	31.0	0	2.78	160.0	0	11.85	137.0	0	9.11
	2	931.0	0	13.63	25441.0	0	503.72	18633.0	320	229.17
	3	13951.0	9	79.68	–	–	–	–	–	–

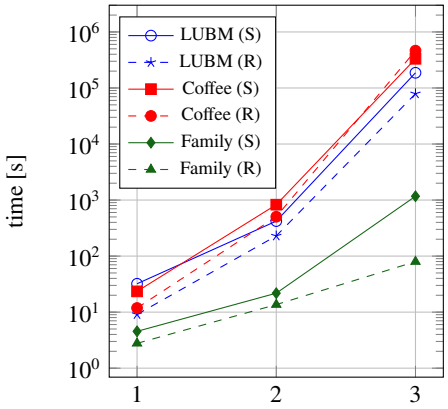

Fig. 1. Depth vs. time

For each run of the algorithm, a number of pruned nodes, reused models and TA calls was recorded. In Figure 2 a proportion of these numbers for each experiment is plotted (for depths for which memory was not exceeded).

The execution time is always higher for the splitting approach than for the reduction approach (ignoring the out-of-memory cases). Indeed this is because in the splitting approach the length limit is applied to each subproblem separately. Thus, also some explanations longer than the limit are computed (as unions of the separate results obtained for each subproblem).

From one point of view, the reduction approach is more efficient, as for a length limit l it always assures all the explanations up to l, and in our experiments it always reached lower time than the splitting approach for the same limit l.

On the other hand, we observed that the splitting approach may often find higher numbers of explanations much more quickly, e.g. in Table 3 (Family ontology). AAA_R found all 9 explanations up to length 3 in 79.68 seconds. AAA_S found 7 of these in 4.54 seconds (after depth 1) and it already found 144 explanations in 21.78 seconds (after depth 2). In fact, it took slightly longer to run it to depth 3 (19.48 minutes) but during this time it found the 9 explanations up to the length 3 together with additional 804 longer explanations. Though we cannot characterize this additional explanations in any way (apart from being sound), this approach may be suitable for some applications, where completeness is not a high priority, and the main goal is to compute as much explanations as possible.

6 Conclusions

We have described a sound and complete ABox abduction algorithm that handles multiple observations, and supports the class of explanations including atomic and negated atomic concept and role assertions. Unlike the previous works which can only support (or are complete up to) limited DL expressivity [18,12,11,19], our algorithm plugs in a DL reasoner as a black box, hence the DL expressivity is only limited by the used

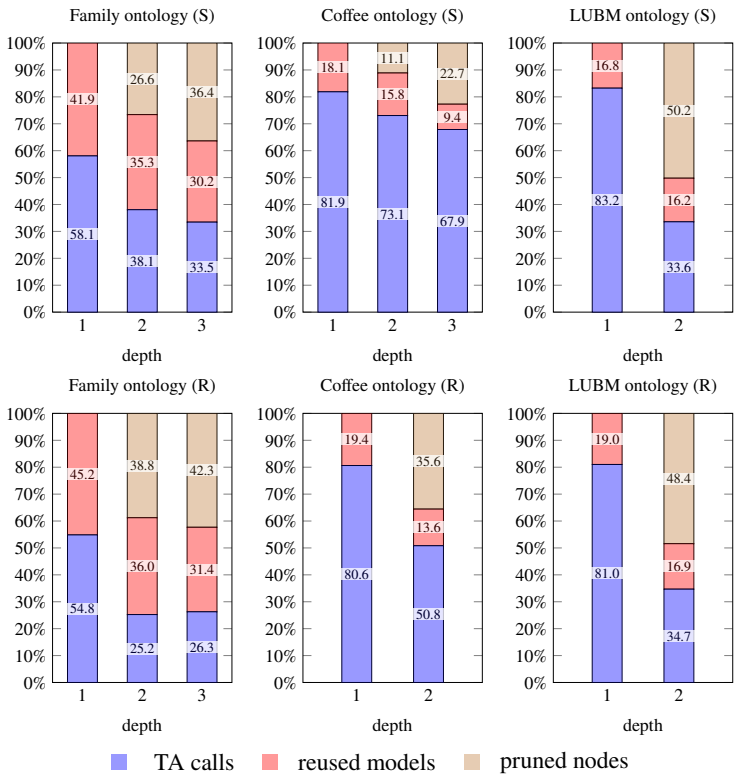

Fig. 2. Proportion of pruned nodes, reused models and TA calls

reasoner. We have provided an implementation based on Pellet [25], thus the expressivity ranges up to $\mathcal{SROIQ}$. MHS is NP-complete [23,17] hence the overall combination inherits the higher complexity of the used DL starting at ExpTime for $\mathcal{ALCHO}$ [13].

In this paper we have predominantly focused on extending our algorithm and implementation to handle multiple-observation abduction problems. We have developed and evaluated two distinct approaches. Our evaluation shows applicability of the approach, particularly when looking for explanations of lower lengths (i.e. those most preferred) and with a smaller number of involved individuals. The evaluation also shows how the implemented optimization techniques are essential in reducing the search space. Note that this is only our first attempt at an implementation, in the future we would like to plug in and compare different reasoners, but also to explore possible improvements in the MHS algorithm [8,27] to further boost the performance.

Acknowledgments. This work was supported by the Slovak national project VEGA 1/0778/18. Júlia Pukancová is also supported by an extraordinary scholarship awarded by Faculty of Mathematics, Physics, and Informatics, Comenius University in Bratislava, and by the Comenius University grant no. UK/266/2018. We would like to thank to anonymous reviewers from previous DL workshops for valuable suggestions.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
2. Castano, S., Espinosa Peraldí, I.S., Ferrara, A., Karkaletsis, V., Kaya, A., Möller, R., Montanelli, S., Petasis, G., Wessel, M.: Multimedia interpretation for dynamic ontology evolution. Journal of Logic and Computation 19(5), 859–897 (2009)
3. Del-Pinto, W., Schmidt, R.A.: Forgetting-based abduction in $\mathcal{ALC}$. In: Proceedings of the Workshop on Second-Order Quantifier Elimination and Related Topics (SOQE 2017), Dresden, Germany. CEUR-WS, vol. 2013, pp. 27–35 (2017)
4. Du, J., Qi, G., Shen, Y., Pan, J.Z.: Towards practical ABox abduction in large description logic ontologies. Int. J. Semantic Web Inf. Syst. 8(2), 1–33 (2012)
5. Elsenbroich, C., Kutz, O., Sattler, U.: A case for abductive reasoning over ontologies. In: Proceedings of the OWLED*06 Workshop on OWL: Experiences and Directions, Athens, GA, USA. CEUR-WS, vol. 2016 (2006)
6. Espinosa Peraldí, I.S., Kaya, A., Möller, R.: Formalizing multimedia interpretation based on abduction over description logic aboxes. In: Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford, UK. CEUR-WS, vol. 477 (2009)
7. Glimm, B., Horrocks, I., Motik, B., Stoilos, G., Wang, Z.: Hermit: An OWL 2 reasoner. Journal of Automated Reasoning 53(3), 245–269 (2014)
8. Greiner, R., Smith, B.A., Wilkerson, R.W.: A correction to the algorithm in reiter's theory of diagnosis. Artificial Intelligence 41(1), 79–88 (1989)
9. Guo, Y., Pan, Z., Heflin, J.: LUBM: A benchmark for OWL knowledge base systems. Journal of Web Semantics 3(2-3), 158–182 (2005)
10. Haarslev, V., Hidde, K., Möller, R., Wessel, M.: The RacerPro knowledge representation and reasoning system. Semantic Web 3(3), 267–277 (2012)
11. Halland, K., Britz, K.: Abox abduction in $\mathcal{ALC}$ using a DL tableau. In: 2012 South African Institute of Computer Scientists and Information Technologists Conference, SAICSIT '12, Pretoria, South Africa. pp. 51–58 (2012)
12. Halland, K., Britz, K.: Naïve ABox abduction in $\mathcal{ALC}$ using a DL tableau. In: Proceedings of the 2012 International Workshop on Description Logics, DL 2012, Rome, Italy. CEUR-WS, vol. 846 (2012)
13. Hladik, J., Model, J.: Tableau systems for $\mathcal{SHIO}$ and $\mathcal{SHIQ}$. In: Haarslev, V., Möller, R. (eds.) Proc. of the 17th Int. Workshop on Description Logics (DL 2004). CEUR-WS, vol. 104, pp. 168–177 (2004)
14. Horrocks, I.: The fact system. In: Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX '98, Oisterwijk, The Netherlands, Proceedings. LNCS, vol. 1397, pp. 307–312. Springer (1998)
15. Horrocks, I.: Using an expressive description logic: FaCT or fiction? In: Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Trento, Italy. pp. 636–649. AAAI (1998)
16. Hubauer, T., Legat, C., Seitz, C.: Empowering adaptive manufacturing with interactive diagnostics: A multi-agent approach. In: Advances on Practical Applications of Agents and Multiagent Systems – 9th International Conference on Practical Applications of Agents and Multiagent Systems, PAAMS 2011, Salamanca, Spain. pp. 47–56 (2011)
17. Karp, R.M.: Reducibility among combinatorial problems. In: Proceedings of a symposium on the Complexity of Computer Computations, IBM Thomas J. Watson Research Center, Yorktown Heights, New York. pp. 85–103 (1972)

18. Klarman, S., Endriss, U., Schlobach, S.: ABox abduction in the description logic $\mathcal{ALC}$. Journal of Automated Reasoning 46(1), 43–80 (2011)
19. Ma, Y., Gu, T., Xu, B., Chang, L.: An ABox abduction algorithm for the description logic $\mathcal{ALCI}$. In: Intelligent Information Processing VI – 7th IFIP TC 12 International Conference, IIP 2012, Guilin, China. Proceedings. IFIP AICT, vol. 385, pp. 125–130. Springer (2012)
20. Peirce, C.S.: Deduction, induction, and hypothesis. Popular science monthly 13, 470–482 (1878)
21. Pukancová, J., Homola, M.: Abductive reasoning with description logics: Use case in medical diagnosis. In: Proceedings of the 28th International Workshop on Description Logics (DL 2015), Athens, Greece. CEUR-WS, vol. 1350 (2015)
22. Pukancová, J., Homola, M.: Tableau-based ABox abduction for the $\mathcal{ALCHO}$ description logic. In: Proceedings of the 30th International Workshop on Description Logics, Montpellier, France. CEUR-WS, vol. 1879 (2017)
23. Reiter, R.: A theory of diagnosis from first principles. Artificial intelligence 32(1), 57–95 (1987)
24. Shearer, R., Motik, B., Horrocks, I.: Hermit: A highly-efficient OWL reasoner. In: Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions, Karlsruhe, Germany. CEUR-WS, vol. 432 (2008)
25. Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Journal of Web Semantics 5(2), 51–53 (2007)
26. Steigmiller, A., Liebig, T., Glimm, B.: Konclude: system description. Journal of Web Semantics 27, 78–85 (2014)
27. Wotawa, F.: A variant of reiter's hitting-set algorithm. Information Processing Letters 79(1), 45–51 (2001)