ABox abduction algorithm for expressive description logics

Júlia Pukancová * and Martin Homola
Comenius University in Bratislava, Mlynská dolina, 842 42 Bratislava, Slovakia
E-mails: pukancova@fmph.uniba.sk, 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. To improve optimality, we include and even slightly extend the pruning techniques proposed by Reiter. 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: Abduction, description logics

1. Introduction

Abduction as form of reasoning was first described by Peirce [24]. Its goal 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 $\mathcal{K} \cup \mathcal{E}$. As a non-standard reasoning task abduction has been recently studied also in description logics (DLs) [10]. ABox abduction, i.e. the case when both $\mathcal{O}$ and $\mathcal{E}$ are limited to ABox assertions, has found applications in areas such as diagnostic reasoning [10, 20, 27] or multimedia interpretation [3, 11].

ABox abduction is useful, but general-purpose ABox abduction solvers, especially for more expressive DLs are still underdeveloped. Some approaches are based on translation, where the abductive task is computed in the target formalism. Klarman et al. [22] proposed an ABox abduction algorithm based on translation to first-order and modal logic. This work is purely theoretical, it is sound and complete, and the expressivity is limited to $\mathcal{ALC}$. Du et al. [8] proposed an approach based on a translation, exploiting an existing Prolog-based abduction solver. They have shown interesting computational results; their approach is sound but it is only complete w.r.t. a specific Horn fragment of $\mathcal{SHIQ}$. Other works [15, 23] exploit directly the tableau reasoning algorithm for DLs, however 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, the soundness or completeness is not shown. Del-Pinto and Schmidt [6] present an abduction solver based on forgetting. Their work includes an implementation, it is sound and complete for $\mathcal{ALC}$.

We present an ABox abduction algorithm building on the ideas of Halland and Britz [15, 16]. It is based on Reiter's [28] Minimal Hitting Set (MHS) algorithm, and it uses a DL reasoner as black box. The algorithm supports atomic and negated atomic concept and role assertions in explanations. It handles abduction problems with multiple observations in form of any ABox assertions. The algorithm exploits optimization techniques suggested by Reiter such as model reuse and pruning. It computes subset-minimal explanations, and shorter explanations are returned first. Thus the search space is explored effectively, starting from more desired explanations. Our work constitutes an extension

*Corresponding author. E-mail: pukancova@fmph.uniba.sk.

of the current state of the art, as it combines the following contributions: (a) expressivity up to OWL 2, i.e. $\mathcal{SROIQ}$; (b) soundness and completeness w.r.t. any given length of explanation; (c) inclusion of model reuse and all applicable Reiter's pruning conditions, which are further extended to exclude more undesired explanations from the search; (d) we provide an implementation (on top of Pellet [30]).

An empirical evaluation on three different ontologies has showed our approach to be feasible, especially when searching for explanations of lower length. We have also showed that the implemented optimization techniques help to significantly reduce the search space, and an interesting comparison between two different approaches to compute multiple observations.

2. Description logics

While our algorithms are complete w.r.t. any DL up to $\mathcal{SROIQ}$ [19] which is the highest expressivity handled by the underlying reasoner, for brevity we will only introduce the $\mathcal{ALCHO}$ DL [cf. 1]. This DL 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 $N_{\mathrm{I}} = \{a, b, c, \dots\}$, set of atomic concepts $N_{\mathrm{C}} = \{A, B, \dots\}$, and set of roles $N_{\mathrm{R}} = \{R, S, \dots\}$. (Complex) $\mathcal{ALCHO}$ *concepts* are recursively constructed starting from atomic concepts and using any of the constructors as stated in Table 1, where C, D are any $\mathcal{ALCHO}$ concepts, R, S are roles, and a, b are individuals.

A TBox $\mathcal{T}$ is a finite set of GCIs, an RBox $\mathcal{R}$ is a finite set of RIAs, and an ABox $\mathcal{A}$ is a finite set of concept assertions, role assertions, and negated role assertions, as given in Table 1, where similarly C, D are any $\mathcal{ALCHO}$ concepts, R, S are roles, and a, b are individuals. These three parts form a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$.

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 $\mathcal{A}$. There is a *clash* in an ABox $\mathcal{A}$ if $\varphi \in \mathcal{A}$ and $\neg\varphi \in \mathcal{A}$.

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

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

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 ($\mathcal{I} \models \varphi$) as given in Table 1.

An interpretation $\mathcal{I}$ 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 $\mathcal{K}$ (denoted by $\mathcal{K} \models \varphi$) if for every model $\mathcal{I}$ of $\mathcal{K}$ it holds that $\mathcal{I} \models \varphi$. A set of axioms Φ is entailed by $\mathcal{K}$ ($\mathcal{K} \models \Phi$) if $\mathcal{K} \models \varphi$ for all $\varphi \in \Phi$.

Entailment and consistency checking are well known to be inter-reducible [1]. Specifically, for any ABox assertion φ, $\mathcal{K} \models \varphi$ if and only if $\mathcal{K} \cup \{\neg\phi\}$ is inconsistent. There is a number of DL reasoners [17, 18, 29–31], mainly solving the consistency checking using the tableau algorithm [1].

3. ABox abduction

According to Elsenbroich et al. [10], in DL we distinguish between TBox and ABox abduction.

Definition 1 (Abduction problem). An *abduction problem* is a pair $\mathcal{P} = (\mathcal{K}, \mathcal{O})$, where $\mathcal{K}$ is a DL knowledge base and $\mathcal{O}$ is set of axioms. A set of axioms $\mathcal{E}$ is an *explanation* of $\mathcal{P}$ if $\mathcal{K} \cup \mathcal{E} \models \mathcal{O}$. Moreover, $\mathcal{P}$ is called

1. $\mathcal{P}$ a *TBox abduction problem*, if $\mathcal{O}$ and $\mathcal{E}$ are limited to TBox axioms;
2. $\mathcal{P}$ an *ABox abduction problem*, if $\mathcal{O}$ and $\mathcal{E}$ are limited to ABox assertions.

If $\mathcal{O}$ and $\mathcal{E}$ are not limited in any way, we also sometimes call $\mathcal{P}$ a knowledge base abduction problem. Such general abduction problems as much as TBox abduction problems are not of our interest in this paper; instead we concentrate on ABox abduction problems. Hence also whenever we say just abduction problem, we mean an ABox abduction problem from here on. We will further differentiate a special case, when $\mathcal{O}$ contains just a sole observation and the general case when it contains more than one observation.

Definition 2 (Single-observation and multiple-observation abduction problems). An abduction problem $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ is called

1. a *single-observation* abduction problem if $\mathcal{O} = \{O\}$ contains only one observation O.
2. a *multiple-observation* abduction problem otherwise.

By an abuse of notation, whenever we write $\mathcal{P} = (\mathcal{K}, O)$ we mean the single-observation abduction problem $\mathcal{P} = (\mathcal{K}, \{O\})$.

Definition 1 provides the basic characterization of an explanation to an abduction problem, however not all explanations are equally acceptable [10]. There are some trivial cases, that we want to rule out.

Definition 3 (Consistent, relevant, and explanatory explanations). Given an ABox abduction problem $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ and its explanation $\mathcal{E}$ we say that:

1. $\mathcal{E}$ is *consistent* if $\mathcal{E} \cup \mathcal{K} \not\models \perp$, i.e. $\mathcal{E}$ is consistent w.r.t. $\mathcal{K}$;
2. $\mathcal{E}$ is *relevant* if $\mathcal{E} \not\models O_i$ for each $O_i \in \mathcal{O}$, i.e. $\mathcal{E}$ does not entail each O_i;
3. $\mathcal{E}$ is *explanatory* if $\mathcal{K} \not\models \mathcal{O}$, i.e. $\mathcal{K}$ does not entail $\mathcal{O}$.

An explanation should be consistent, as anything follows from inconsistency; and so, an explanation that makes $\mathcal{K}$ inconsistent does not really explain the observation. It should be relevant, that is, it should not imply the observation directly without requiring the knowledge base $\mathcal{K}$ at all. And it should be explanatory, that is, we should not be able to explain the observation without it.

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, sometimes also called subset-minimality, already employed by [28].

Definition 4 (Syntactic minimality). Given an ABox abduction problem $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ and two explanations $\mathcal{E}$ and $\mathcal{E}'$ of $\mathcal{P}$, we say that $\mathcal{E}$ is *smaller* than $\mathcal{E}'$ if $\mathcal{E} \subsetneq \mathcal{E}'$.We further say that a solution $\mathcal{E}$ of $\mathcal{P}$ is *syntactically minimal* if there is no other solution $\mathcal{E}'$ of $\mathcal{P}$ that is smaller than $\mathcal{E}$.

This notion of minimality, based on subsets is not the only one which has been considered in literature. *Cardinality-based minimality* [9, 25]. considers as minimal only those explanations $\mathcal{E}$ such that for all other explanations $\mathcal{E}'$ we have $|\mathcal{E}| \leq |\mathcal{E}'|$. Although syntactic minimality will be our main focus in this paper, our algorithms can also be used to compute cardinality-minimal explanations.

Further, semantic minimality [5] considers explanations which cannot be logically implied by other explanations. Given two solutions $\mathcal{E}$ and $\mathcal{E}'$ of $\mathcal{P} = (\mathcal{K}, \mathcal{O})$, we say that $\mathcal{E}$ is (semantically) *weaker* than $\mathcal{E}'$ (denoted by $\mathcal{E}' \prec_{\mathcal{K}} \mathcal{E}$) if $\mathcal{K} \cup \mathcal{E}' \models \mathcal{E}$ but not $\mathcal{K} \cup \mathcal{E} \models \mathcal{E}'$. In such a case $\mathcal{E}'$ is also said to be (semantically) stronger than $\mathcal{E}$. A solution $\mathcal{E}$ of $\mathcal{P}$ is *semantically minimal* if there is no $\mathcal{E}'$ s.t. $\mathcal{E} \prec_{\mathcal{K}} \mathcal{E}'$. We do not consider semantic minimality in this work and it is rarely considered in DL abduction research. For instance [6] applies this notion of minimality. Notably, while in diagnostic applications of abduction one would expect weaker explanations to be preferred [27], in the work of [26] who apply abduction on multimedia interpretation semantically stronger explanations are of interest.

In this work we are interested in explanations in form of atomic and negated atomic ABox assertions (which satisfy the requirements of Definitions 3–4). Let us now define this class of observations formally.

Definition 5 ($\mathbf{A_nR_n}$ and $\mathbf{A_nR_n^{CER,sub}}$). Given an abduction problem $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ we define the following classes of explanations:

1. $\mathrm{A_nR_n}(\mathcal{P})$ contains all explanations $\mathcal{E}$ of $\mathcal{P}$ such that $\mathcal{E} \subseteq \{A(a), \neg A(a), R(a, b), \neg R(a, b) \mid A \in N_{\mathrm{C}}, R \in N_{\mathrm{R}}, a, b \in N_{\mathrm{I}}\}$;
2. $\mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$ contains all explanations $\mathcal{E}$ of $\mathcal{P}$ such that $\mathcal{E} \in \mathrm{A_nR_n}$ and $\mathcal{E}$ is explanatory, consistent, relevant, and minimal.

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

4. Explaining a single observation

To stay within the class $\mathrm{A_nR_n}$ (or, in fact, its subclass $\mathrm{A_nR_n^{CER,sub}}$) we need to be able to extract all suitable ABox assertions from the models computed by the tableau algorithm. This will be done by so called ABox encoding of models.

Definition 6 (ABox encoding). The *ABox encoding* of an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ is:

$$\begin{aligned} M_\mathcal{I} = \{C(a) \mid \mathcal{I} \models C(a), C \in \{A, \neg A\}, A \in N_\mathrm{C}, a \in N_\mathrm{I}\} \\ \cup \{R(a,b) \mid \mathcal{I} \models R(a,b),\ R \in N_\mathrm{R},\ a, b \in N_\mathrm{I}\} \\ \cup \{\neg R(a,b) \mid \mathcal{I} \models \neg R(a,b),\ R \in N_\mathrm{R},\ a, b \in N_\mathrm{I}\}\,. \end{aligned}$$

A DL reasoner called by our abduction algorithm as a black box is simply represented by a function TA() such that $\mathrm{TA}(\mathcal{K}) = M_\mathcal{I}$ returns the ABox encoding of some model $\mathcal{I}$ of $\mathcal{K}$ if $\mathcal{K}$ is consistent, otherwise $\mathrm{TA}(\mathcal{K}) = \{\}$.

Similarly as above, each $\mathcal{K}$ and O have a finite combined signature, hence each ABox encoding is finite. What is more, observe that $M_\mathcal{I}$ is in no way 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.

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. As suggested by Halland and Britz [15, 16], such $\mathcal{E}$ corresponds to a set of ABox assertions which causes that no model of $\mathcal{K} \cup \{\neg O\}$ will be a model of $\mathcal{K} \cup \mathcal{E} \cup \{\neg O\}$ anymore. As observed by Reiter [28], such a set $\mathcal{E}$ can be found as a *hitting set* [21] of the collection $\mathcal{M}$ of all models of $\mathcal{K} \cup \{\neg O\}$ and then negating each assertion in the hitting set.

Definition 7 (Hitting Set). Given a set of sets $\mathcal{M}$, a hitting set H of $\mathcal{M}$ is any set such that $H \cap M \neq \{\}$ for every $M \in \mathcal{M}$.

In other words, it sufficed to find one ABox assertion from the ABox encoding of each model and add its negation into $\mathcal{E}$.

To find all hitting sets for a collection of sets $\mathcal{M}$, Reiter [28] proposed to construct a hitting set tree (HS-tree). In his HS-tree nodes are labelled by a set from $\mathcal{M}$ or by ✓ (although we will use $\{\}$). Edges from a node n are labelled by an element from the label of n. Additional constraints ensure that the edge-labels on every path from the root to a leaf correspond to a hitting set.

Definition 8 (HS-tree). An HS-tree for a collection of sets $\mathcal{M} \neq \{\}$ is a smallest labelled tree $T = (V, E, L)$ with root r, nodes and edges labelled by $L()$, and $H(n)$ being the set of edge-labels on the path from r to a node n, such that:

1. $L(r) = M$ for some $M \in \mathcal{M}$.
2. If $L(n) = \{\}$, it has no successors in T.
3. If $L(n) = M_n \in \mathcal{M}$, then for each $\sigma \in M_n$, n has a successor n_σ, s.t. $L(n, n_\sigma) = \sigma$ and $L(n_\sigma) = M_{n_\sigma} \in \mathcal{M}$ s.t. $M_{n_\sigma} \cap H(n_\sigma) = \{\}$ if it exists, or $L(n_\sigma) = \{\}$ otherwise.

The HS-tree respective to any empty set $\mathcal{M} = \{\}$ is intentionally undefined.

To compute only (subset) minimal hitting sets, Reiter proposed to construct the HS-tree by breadth-first search and to prune paths that would not lead to minimal hitting sets. This is particularly useful, as we are only interested in minimal explanations, which correspond to minimal hitting sets. In addition, we want to filter out explanations which are not explanatory, consistent, and relevant. We can do this by extending the pruning conditions originally proposed by Reiter. All nodes will be labelled either by $M \in \mathcal{M}$, or by $\{\}$, or by $\times$. The latter two labels will be used to steer the pruning.

Definition 9 (Pruned node). Given an abduction problem $\mathcal{P} = (\mathcal{K}, O)$, let $\mathcal{M} = \mathrm{TA}(\mathcal{K} \cup \{\neg O\})$ and let $T = (V, E, L)$ be a HS-tree for a collection of sets $\{\neg M \mid M \in \mathcal{M}\}$. A node $n \in V$ in T for is pruned if:

(a) either there is $n' \in V$ s.t. $H(n') \subseteq H(n)$ and $L(n') = \{\}$
(label n by $\{\}$);
(b) or there is $n' \in V$ s.t. $H(n') = H(n)$ and $L(n') = M \in \mathcal{M}$
(label n by $\times$);
(c) or $\{\neg O\} \cup H(n)$ is inconsistent (label n by $\{\}$);
(d) or $H(n) \cup \mathcal{K}$ is inconsistent (label n by $\{\}$);

Condition (a) represents Reiter's first pruning condition, i.e., when another node n', s.t. $H(n') \subseteq H(n)$ corresponds to a smaller hitting set. However, we extend this condition also to cases when such n' corresponds to a node previously pruned according to (c) or (d).

Condition (b) corresponds to Reiter's second pruning condition, i.e., when another node n' has the same $H(n') = H(n)$. There is no need to continue in the doubled path, and so one of them is pruned – node n is labelled by $\times$.

Note that Reiter's third pruning condition never applies in our case because for no two nodes $L(n) \subseteq L(n')$, due to node labels corresponding to ABox encodings of models.

Conditions (c) and (d) are new, and they are responsible for pruning all paths respective to irrelevant and inconsistent explanations (in conjunction with condition (a) which also prunes all supersets of such previously found paths).

Definition 10 (Pruned HS-tree). A pruned HS-tree is obtained from a HS-tree by removing all pruned nodes including their descendants.

Reiter [28] has proven, that in a pruned HS-tree, the paths from the root to the leaves that are not pruned correspond exactly to all minimal hitting sets. While it was later showed by Greiner et al. [12] that this result is not entirely correct, the problem lies in Reiter's third pruning condition which does not apply in our case.

But since we have extended the first two of Reiter's original pruning conditions taking into account additional cases in which are irrelevant or inconsistent, his result now extends as follows.

Theorem 1. *Given an abduction problem $\mathcal{P} = (\mathcal{K}, O)$, let $\mathcal{M} = \mathrm{TA}(\mathcal{K} \cup \{\neg O\})$ and let $T = (V, E, L)$ be a pruned HS-tree for a collection of sets $\{\neg M \mid M \in \mathcal{M}\}$. Then $\{H(n) \mid n \in V,\ L(n) = \{\}$, and n is not pruned$\} = \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$.*

Proof. Let us remind, that all the pruning conditions proposed by Reiter except the last one are included in our pruning. The Reiter's last condition to prune deals with two nodes n_1 and n_2 labelled as follows $L(n_1) = S_1 \in \mathcal{M}$ and $L(n_2) = S_2 \in \mathcal{M}$, while $S_1 \subsetneq S_2$. This situation never appears in our approach, as our collection $\mathcal{M}$ contains only sets containing each atomic ABox assertion from $\mathcal{K}$ either in positive way or its complement, and thus for no $S_1, S_2 \in \mathcal{M}$ it can hold that $S_1 \subsetneq S_2$. Since this condition never applies, and the other Reiter's conditions are included in our approach, only minimal hitting sets are found.

The additional conditions (c) and (d) only filter from these all minimal hitting sets such sets, that are irrelevant and inconsistent. In addition, also supersets of such previously pruned sets are filtered out, as part of condition (a). That is, in our HS-tree, all minimal hitting sets corresponding to the relevant and consistent explanations are found.

Recall also that, an explanation $\mathcal{E}$ for $\mathcal{P} = (\mathcal{K}, O)$ is explanatory if and only if $\mathcal{K} \not\models O$, i.e. $\mathcal{K} \cup \{\neg O\}$ has at least one model. Hence, if there is a HS-tree with a non-empty collection of minimal hitting sets, then all these minimal hitting sets must correspond to explanatory explanations. □

We first introduce an abduction algorithm that handles a special case with just one observation, i.e., the Single Observation Algorithm (SOA). It is listed in Algorithm 1. It takes five inputs. A DL knowledge base $\mathcal{K}$, a single observation O in form of any ABox assertion (concept or role), and an upper bound $l \geq 1$ on the maximal explanation length, are the first three. The last two inputs are auxiliary and they are utilized in the case when SOA is reused to compute multiple observations. $\mathcal{O}$ is the overall set of observations, and s_0 is an auxiliary individual that is to be ignored in the explanations. For now let us assume that $\mathcal{O} = \{O\}$ and s_0 does not appear in $\mathcal{K}$ and O.

The algorithm first checks if $\mathcal{K}' = \mathcal{K} \cup \{\neg O\}$ has at least one model M (calling TA), because otherwise there is nothing to explain and the algorithm immediately terminates (lines 1–5).

In lines 7–10 the root of the HS-tree is initialized and labelled by M, together with its successors for each $\sigma \in M$.

The algorithm then traverses all nodes n by breadth-first search and recursively extends the HS-tree while applying both the model-reuse and the pruning optimization techniques. First, pruning conditions (a), (b), (c), and (by checking for a clash in $H(n)$) partially also (d) are tested (lines 14–17). We can afford this as these are computationally cheap operations.

Testing full condition (d) is postponed, as it involves an expensive TA call for $\mathcal{K} \cup H(n)$. Before we do this, we check if a model for $\mathcal{K}' \cup H(n)$ is found among those stored for reuse (line 19; note that this rules out pruning condition (d)).

If no model can be reused we call TA for $\mathcal{K}' \cup H(n)$ (line 21) to obtain a new model M. If there is one, n is labelled by $\neg M$ and its successors are added to the HS-tree. Otherwise we have found an explanation $H(n)$. Thanks to the pruning applied so far, this explanation is minimal, explanatory and relevant, but consistence has to be still checked. If $H(n)$ is not consistent, the node is pruned (pruning condition (d), line 28), otherwise the explanation is stored.

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 $\mathcal{E} \in \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$ s.t. $|\mathcal{E}| \leq l$

1: $\mathcal{K}' \leftarrow \mathcal{K} \cup \{\neg O\}$
2: $M \leftarrow \mathrm{TA}(\mathcal{K}')$
3: **if** $M = \{\}$ **then**
4:     **return** `"nothing to explain"`
5: **end if**
6: $M \leftarrow M \setminus \{\varphi \mid \varphi \in M$ and s_0 occurs in $\varphi\}$
7: $MS \leftarrow \{M\}$ ▷ store M for reuse
8: create new HS-tree $T = (V, E, L)$ with root r
9: $L(r) \leftarrow \neg M$
10: for each $\sigma \in \neg M$ create a successor n_σ of r and label the resp. edge by σ
11: $\mathcal{S}_\mathcal{E} \leftarrow \{\}$
12: $n \leftarrow$ next node w.r.t. r in T by breadth-first search
13: **while** $n \neq$ null **and** $|H(n)| \leq l$ **do**
14:     **if** clash in $H(n)$ **or** ($n' \in V$ **and** $H(n') \subseteq H(n)$ **and** $L(n') = \{\}$) **or** $H(n) \cup \{\neg O_i\}$ is incons. for some $O_i \in \mathcal{O}$ **then**
15:         $M \leftarrow \{\}$ ▷ pruning (a) or (c)
16:     **else if** $n' \in V$ **and** $H(n) = H(n')$ **and** $L(n') \neq$ null
17:         **then** $M \leftarrow \times$ ▷ pruning (b)
18:     **else if** $N \in MS$ **and** $H(n) \subseteq N$ **then**
19:         $M \leftarrow N$ ▷ reuse model
20:     **else**
21:         $M \leftarrow \mathrm{TA}(\mathcal{K}' \cup H(n))$
22:         $M \leftarrow M \setminus \{\varphi \mid \varphi \in M$ and s_0 occurs in $\varphi\}$
23:         **if** $M \neq \{\}$ **then**
24:             $MS \leftarrow MS \cup \{M\}$ ▷ store M for reuse
25:         **else if** $\mathcal{K} \cup H(n)$ is consistent **then**
26:             $\mathcal{S}_\mathcal{E} \leftarrow \mathcal{S}_\mathcal{E} \cup \{H(n)\}$ ▷ store explanation
27:         **else**
28:             $M \leftarrow \{\}$ ▷ pruning (d)
29:         **end if**
30:     **end if**
31:     $L(n) \leftarrow \neg M$
32:     for each $\sigma \in \neg M$ create a successor n_σ of n and label the resp. edge by σ
33:     $n \leftarrow$ next node in T w.r.t. n by breadth-first search
34: **end while**
35: **return** $\mathcal{S}_\mathcal{E}$

Once the breadth-first search is over, the stored explanations are returned on the output.

We will now show that the SOA algorithm is correct w.r.t. the single-observation abduction problem.

Lemma 1 (Soundness)**.** *Let $\mathcal{P} = (\mathcal{K}, O)$ be a single-observation abduction problem, and let $l \geq 1$. Let $\mathcal{S}_\mathcal{E} := SOA(\mathcal{K}, O, l, \{O\}, s_0)$. Then $\mathcal{S}_\mathcal{E} \subseteq \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$.*

Proof. If SOA returned `"nothing to explain"`, it must have terminated in line 4, and this was because $\mathcal{K} \cup \{\neg O\}$ was inconsistent, which is the same as $\mathcal{K} \models O$, and in such a case there are no explanations.

In the other case SOA returned a set $\mathcal{S}_\mathcal{E}$. Let $\mathcal{E} \in \mathcal{S}_\mathcal{E}$. In such a case $\mathcal{E} = H(n)$ for some node n and it was added to $\mathcal{S}_\mathcal{E}$ in line 26. However in this case we have also called TA for $\mathcal{K} \cup \{\neg O\} \cup \mathcal{E}$ in line 21 and it returned no model (as tested in line 23). Hence $\mathcal{K} \cup \mathcal{E} \models O$, i.e., $\mathcal{E}$ is an explanation of $\mathcal{P}$. $\mathcal{E}$ is relevant and consistent, as we have tested in lines 14 and 25, respectively. It is also explanatory because if not the algorithm returned `"nothing to explain"` and terminated already in line 4 as described above.

As we already argued, the algorithm constructs the pruned-HS tree correctly according to Definition 10, as it always extends each node n with successors corresponding to all $\sigma \in L(n)$ if they are not pruned, and it prunes all nodes according to Definition 9. The minimality of $\mathcal{E}$ then follows from the fact that we only add such $\mathcal{E} = H(n)$ into $\mathcal{S}_\mathcal{E}$ in line 26 which correspond to paths from root to a leaf which are not pruned, and according to Theorem 1 in a pruned HS-tree all such paths correspond to minimal hitting sets. □

Lemma 2 (Completeness)**.** *Let $\mathcal{P} = (\mathcal{K}, O)$ be a single-observation abduction problem, and let $l \geq 1$. Let $\mathcal{E} \in \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$ and let $|\mathcal{E}| \leq l$. Let $\mathcal{S}_\mathcal{E} := SOA(\mathcal{K}, O, l, \{O\}, s_0)$. Then $\mathcal{E} \in \mathcal{S}_\mathcal{E}$.*

Proof. Let $\mathcal{S}_\mathcal{E} := \mathrm{SOA}(\mathcal{K}, O, l, \{O\}, s_0)$ and let $\mathcal{E}$ be a consistent, relevant, explanatory, and minimal explanation of $\mathcal{P}$.

We will prove by induction on the depth k, that there is some node n with $|H(n)| = k$ s.t. $H(n) \subseteq \mathcal{E}$, that is not pruned.

The *base case:* let $k = 1$. As $\mathcal{E}$ is explanatory, $\mathcal{K} \cup \{\neg O\}$ has at least one model M. Hence the HS-tree T constructed by SOA has at last one node r, labelled by $\neg M$ in line 9. Note that we use the ABox encoding of M, and hence $\varphi \in \neg M$ or $\neg\varphi \in \neg M$ for every atomic ABox assertion φ.

It is clear that $\mathcal{K} \cup M \cup \{\neg O\}$ is consistent and so $\mathcal{E} \nsubseteq M$, i.e. there is an ABox assertion $\sigma_1 \in \mathcal{E}$ s.t. $\sigma_1 \notin M$. Hence $\neg\sigma_1 \in M$, and so $\sigma_1 \in \neg M$, and also $L(r, n_{\sigma_1}) = \sigma_1$ for some successor n_{σ_1} of r, in line 10.

The *induction step*. Let us assume that SOA extended T until there is a node n_{σ_k} s.t. $H(n_{\sigma_k}) \subseteq \mathcal{E}$ and $|H(n_{\sigma_k})| = k$. We will show that $(*)$ either $H(n_{\sigma_k}) = \mathcal{E}$ or there is some $\sigma_{k+1} \in \mathcal{E} \setminus H(n_{\sigma_k})$ which will become the label of some new edge leading from n_{σ_k}.

Observe that none of the pruning conditions in line 14 applies on n_{σ_k}, as $\mathcal{E}$ is consistent, and hence $H(n_{\sigma_k})$ does not contain a clash; there is no $H(n) \subsetneq H(n_{\sigma_k})$ s.t. $L(n) = \{\}$, because $\mathcal{E}$ is minimal, and $H(n_{\sigma_k}) \cup \{\neg O\}$ is consistent because $\mathcal{E}$ is relevant. Also, if there is some other node n in T such that $H(n) = H(n_{\sigma_k})$ we can assume w.l.o.g. that n_{σ_k} is the one which is visited first and hence it is not pruned in line 17.

Next we distinguish two cases. In the first case $\mathcal{K} \cup H(n_{\sigma_k}) \cup \{\neg O\}$ is inconsistent, and so $H(n_{\sigma_k}) = \mathcal{E}$. As $\mathcal{E}$ is a consistent explanation, $\mathcal{K} \cup H(n_{\sigma_k})$ is also consistent and therefore SOA adds $H(n_{\sigma_k}) = \mathcal{E}$ into $\mathcal{S}_{\mathcal{E}}$ in line 26. In the second case $\mathcal{K} \cup H(n_{\sigma_k}) \cup \{\neg O\}$ is consistent, i.e. it has a model M_k, either reused in line 19 or found by a TA call in line 21, and $L(n_{\sigma_k})$ is set to $\neg M_k$ in line 31. It is clear that $H(n_{\sigma_k}) \subseteq M_k$ and that $\mathcal{K} \cup M_k \cup \{\neg O\}$ is consistent and so $\mathcal{E} \nsubseteq M_k$, i.e. there is $\sigma_{k+1} \in \mathcal{E} \setminus H(n_{\sigma_k})$ s.t. $\sigma_{k+1} \notin M_k$, and so $\sigma_{k+1} \in \neg M_k$. Therefore SOA consequently creates a node $n_{\sigma_{k+1}}$ with $L(n_{\sigma_k}, n_{\sigma_{k+1}}) = \sigma_{k+1}$.

We have proved $(*)$ for any k. By induction SOA will eventually create a node n_{σ_m} s.t. $H(n_{\sigma_m}) = \mathcal{E}$ and $L(n_{\sigma_m}) = \{\}$. Thus SOA finally adds $H(n_{\sigma_m}) = \mathcal{E}$ into $\mathcal{S}_{\mathcal{E}}$. Also, as we construct the HS-tree breadth-first and since $|\mathcal{E}| = |H(n_{\sigma_m})| \leq l$, the while loop in line 13 surely does not terminate before the node n_{σ_m} is visited and fully processed. $\square$

Summing up, the algorithm is correct as it finds exactly all desired explanations up to the given length l. It also terminates, as the HS-tree construction is depth-bound and there are only finitely many ABox assertions that may serve as labels of successor edges starting from any node of the tree.

Theorem 2. *Let $\mathcal{P} = (\mathcal{K}, O)$ be a single-observation abduction problem, and $l \geq 1$. Then SOA($\mathcal{K}, O, l, \{O\}, s_0$) always terminates, and it is sound and complete w.r.t. all $\mathcal{E} \in \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$ s.t. $|\mathcal{E}| \leq l$.*

In addition, if we remove the depth 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 abduction problem.

Corollary 1. *Let $\mathcal{P} = (\mathcal{K}, O)$ be a single-observation abduction problem. Then SOA($\mathcal{K}, O, \infty, \{O\}, s_0$) always terminates, and it is sound and complete w.r.t. $\mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$.*

Computationally, the hitting set problem is NP-complete [21]. The MHS algorithm constructs the HS-tree in exponential time, more precisely $O(b^d)$ where b is the branching factor bound (i.e., the number of sets in $\mathcal{M}$) and d is maximum depth of the HS-tree (i.e., the number of elements in $\bigcup \mathcal{M}$). This brings us the following result.

Theorem 3. *The worst-case time complexity of SOA is $O(n^l) \cdot X$ when the observations are bound by the maximal length l and $O(n^n) \cdot X$ in the unbounded case, where $n = b \cdot a + c \cdot a^2$, $a = |N_\mathrm{I}|$, $b = |N_\mathrm{C}|$, $c = |N_\mathrm{R}|$, and X is the worst-case time complexity of the reasoner called as a black box.*

Note that in this estimate we again assume the finite vocabulary respective to the finite knowledge base $\mathcal{K}' = \mathcal{K} \cup \mathcal{O}$. We can also relate this complexity to the size of the input $\mathcal{K}'$ that we will denote m. As in the worst case n is at most polynomial with respect to m, we obtain that the worst-case time complexity of SOA is $O(2^{poly(m)}) \cdot X$. Hence, assuming that we will use an optimal solver for a given DL we obtain the following corollary.

Corollary 2. *The SOA algorithm is in* ExpTime *for all DLs up to $\mathcal{SHOI}$, $\mathcal{SHIQ}$, and $\mathcal{SHOQ}$. It is in* NExpTime *for $\mathcal{SHOIQ}$, and it is in* N2ExpTime *for $\mathcal{SROIQ}$.*

5. Explaining multiple observations

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

5.1. 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, however in DLs featuring nominals, they may be encoded as follows.

Lemma 3 (Reduction)**.** *Let $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ be a multiple-observation abduction problem with $\mathcal{O} = \{C_1(a_1), \dots, C_n(a_n), R_1(b_1, c_1), \dots, R_m(b_m, c_m), \neg Q_1(d_1, e_1), \dots, \neg Q_l(d_l, e_l)\}$. 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*

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

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

Proof. *Only-if part.* By contradiction, assume that $\mathcal{E}$ is an explanation of $\mathcal{P}$, but not of $\mathcal{P}'$. Then $\mathcal{K} \cup \mathcal{E} \not\models O'$, hence $\mathcal{K} \cup \mathcal{E} \cup \{\neg X(s_0)\}$ has a model $\mathcal{I}$. Note that

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

This means that $s_0^{\mathcal{I}}$ either belongs to $\{a_i\} \sqcap \neg C_i^{\mathcal{I}}$ for some $i \in [1..n]$ and hence also $a_i^{\mathcal{I}} \in \neg C_i^{\mathcal{I}}$ and thus $a^{\mathcal{I}} \notin C_i^{\mathcal{I}}$; or $s_0^{\mathcal{I}}$ belongs to $\{b_j\} \sqcap \forall R_j.\neg\{c_j\}^{\mathcal{I}}$ for some $j \in [1..m]$ and hence also $b_j^{\mathcal{I}} \in \forall R_j.\neg\{c_j\}^{\mathcal{I}}$ and thus $(b_j^{\mathcal{I}}, c_j^{\mathcal{I}}) \notin R_j^{\mathcal{I}}$; or $s_0^{\mathcal{I}}$ belongs to $\{d_k\} \sqcap \exists Q_k.\{e_k\}^{\mathcal{I}}$ for some $k \in [1..l]$ and hence also $d_k^{\mathcal{I}} \in \exists Q_k.\{e_k\}^{\mathcal{I}}$ and thus $(d_k^{\mathcal{I}}, e_k^{\mathcal{I}}) \in Q_k^{\mathcal{I}}$. In the first case we have $\mathcal{I} \not\models C_i(a_i)$; in the second case we have $\mathcal{I} \not\models R_j(b_j, c_j)$; and in the third case we have $\mathcal{I} \not\models \neg Q_k(d_k, e_k)$. Either case contradicts $\mathcal{E}$ being an explanation of $\mathcal{P}$.

If part. By contradiction, assume that $\mathcal{E}$ is an explanation of $\mathcal{P}'$, but not of $\mathcal{P}$. Then either $\mathcal{K} \cup \mathcal{E} \not\models C_i(a_i)$ for some $i \in [1..n]$, or $\mathcal{K} \cup \mathcal{E} \not\models R_j(b_j, c_j)$ for some $j \in [1..m]$, or $\mathcal{K} \cup \mathcal{E} \not\models \neg Q_k(d_k, e_k)$ for some $k \in [1..l]$.

In the first case, $\mathcal{K} \cup \mathcal{E} \cup \{\neg C_i(a_i)\}$ has a model $\mathcal{I}$. Let $\mathcal{I}'$ be $\mathcal{I}$ extended with $s_0^{\mathcal{I}} = a_i^{\mathcal{I}}$. Now, $\mathcal{I}'$ is a model of $\mathcal{K} \cup \mathcal{E} \cup \{\{a_i\} \sqcap \neg C_i(s_0)\}$ and thus also of $\mathcal{K} \cup \mathcal{E} \cup \{\neg X(s_0)\}$. Thus $\mathcal{K} \cup \mathcal{E} \not\models X(s_0)$, that is, $\mathcal{K} \cup \mathcal{E} \not\models O'$ which is a contradiction.

In the second case, $\mathcal{K} \cup \mathcal{E} \cup \{\neg R_j(b_j, c_j)\}$ has a model $\mathcal{I}$. Let $\mathcal{I}'$ be $\mathcal{I}$ extended with $s_0^{\mathcal{I}} = b_j^{\mathcal{I}}$. Now, $\mathcal{I}'$ is a model of $\mathcal{K} \cup \mathcal{E} \cup \{\{b_j\} \sqcap \forall R_j.\neg\{c_j\}(s_0)\}$ and thus also of $\mathcal{K} \cup \mathcal{E} \cup \{\neg X(s_0)\}$. Thus again $\mathcal{K} \cup \mathcal{E} \not\models X(s_0)$, that is, $\mathcal{K} \cup \mathcal{E} \not\models O'$ which is a contradiction.

In the third case, $\mathcal{K} \cup \mathcal{E} \cup \{Q_k(d_k, e_k)\}$ has a model $\mathcal{I}$. Let $\mathcal{I}'$ be $\mathcal{I}$ extended with $s_0^{\mathcal{I}} = d_k^{\mathcal{I}}$. Now, $\mathcal{I}'$ is a model of $\mathcal{K} \cup \mathcal{E} \cup \{\{d_k\} \sqcap \exists Q_k.\{e_k\}(s_0)\}$ and thus also of $\mathcal{K} \cup \mathcal{E} \cup \{\neg X(s_0)\}$. Thus again $\mathcal{K} \cup \mathcal{E} \not\models X(s_0)$, that is, $\mathcal{K} \cup \mathcal{E} \not\models O'$ which is a contradiction. □

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 $\mathcal{P}$, as shown by the following example. 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 3, 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 3 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 $\mathcal{O} = \{C_1(a_1), \dots, C_n(a_n), R_1(b_1, c_1), \dots, R_m(b_m, c_m), \neg Q_1(d_1, e_1), \dots, \neg Q_l(d_l, e_l)\}$, max length of an explanation l

Ensure: set $\mathcal{S}_{\mathcal{E}}$ of all $\mathcal{E} \in \mathrm{A_n R_n^{CER,sub}}(\mathcal{P})$ s.t. $|\mathcal{E}| \leq l$

$s_0 \leftarrow$ new individual w.r.t. $\mathcal{K}$ and $\mathcal{O}$

$X \leftarrow (\neg\{a_1\} \sqcup C_1) \sqcap \dots \sqcap (\neg\{a_n\} \sqcup C_n)$
$\quad \sqcap (\neg\{b_1\} \sqcup \exists R_1.\{c_1\}) \sqcap \dots \sqcap (\neg\{b_m\} \sqcup \exists R_m.\{c_m\})$
$\quad \sqcap (\neg\{d_1\} \sqcup \forall Q_1.\neg\{e_1\}) \sqcap \dots \sqcap (\neg\{d_l\} \sqcup \forall Q_l.\neg\{e_l\})$

$O' \leftarrow X(s_0)$

$\mathcal{S}_{\mathcal{E}} \leftarrow \mathrm{SOA}(\mathcal{K}, O', l, \mathcal{O}, s_0)$

return $\mathcal{S}_{\mathcal{E}}$

$\mathrm{AAA_R}$ makes use of the auxiliary parameters of SOA. Instead of filtering the unwanted explanation involving the auxiliary individual s_0 ex post, it does so in much more optimal fashion, by passing s_0 to SOA as the fifth parameter. SOA then excludes the assertions involving s_0 already from the models returned by TA, and hence reducing the HS-tree that needs to be constructed.

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.

Theorem 4. *Let $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ be a multiple-observation abduction problem, and let $l \geq 1$. Then AAA_R $(\mathcal{K}, \mathcal{O}, l)$ always terminates, and it is sound and complete w.r.t. all $\mathcal{E} \subseteq \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$ s.t. $|\mathcal{E}| \leq l$.*

The theorem is a direct consequence of Lemma 3 and the construction of Algorithm 2 which calls SOA with an observation reduced into a single assertion, and features only minor modifications whose only effect is filtering out assertions involving the individual s_0 from the candidate explanations.

Observing that SOA handles the auxiliary parameters correctly, the correctness of $\mathrm{AAA_R}$ is then consequence of the correctness of SOA.

Corollary 3. *Given a multiple-observation abduction problem $\mathcal{P} = (\mathcal{K}, \mathcal{O})$, AAA_R $(\mathcal{K}, \mathcal{O}, \infty)$ always terminates, and it is sound and complete w.r.t.* $\mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$.

5.2. *Splitting into subproblems*

Instead of reducing a multiple-observation abduction problem $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ to one with a single observation, we will now show how to solve it by splitting $\mathcal{P}$ into n subproblems $\mathcal{P}_i = (\mathcal{K}_i, O_i)$, answering each $\mathcal{P}_i$ separately using SOA, and then combining the results.

If there is no bound on length, it is quite easy to combine the subproblems: we simply combine the results in terms of union with some additional filtering. 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.

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

The algorithm starts by initializing 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 all observations $\mathcal{O}$ as the fourth parameter due to relevance checks and the dummy s_0. If one of the observations cannot be explained (SOA returned $\{\}$) then neither the whole set $\mathcal{O}$ can be explained: the algorithm returns $\{\}$ and terminates (line 7).

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 $\mathcal{S_E}$ of all $\mathcal{E} \in \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$ s.t. $|\mathcal{E}| \leq l$
1: $s_0 \leftarrow$ new individual w.r.t. $\mathcal{K}$ and $\mathcal{O}$ ▷ auxiliary
2: $\Sigma \leftarrow \{\}$ ▷ collection of partial results
3: $\mathcal{K}' \leftarrow \mathcal{K} \cup \{\top(a) \mid a \text{ occurs in } O_i, O_i \in \mathcal{O}\}$
4: **for all** $O_i \in \mathcal{O}$ **do**
5: $\quad \mathcal{S}_{\mathcal{E}_i} \leftarrow \mathrm{SOA}(\mathcal{K}', O_i, l, \mathcal{O}, s_0)$ ▷ store partial result
6: $\quad$ **if** $\mathcal{S}_{\mathcal{E}_i} = \{\}$ **then**
7: $\quad\quad$ **return** $\{\}$ ▷ $\mathcal{O}$ has no explanation
8: $\quad$ **else if** $\mathcal{S}_{\mathcal{E}_i} \neq$ `"nothing to explain"` **then**
9: $\quad\quad \Sigma \leftarrow \Sigma \cup \{\mathcal{S}_{\mathcal{E}_i}\}$ ▷ store partial result
10: $\quad$ **end if**
11: **end for**
12: **if** $\Sigma = \{\}$ **then**
13: $\quad$ **return** `"nothing to explain"`
14: **else**
15: $\quad \mathcal{S_E} \leftarrow \{\mathcal{E}_1 \cup \cdots \cup \mathcal{E}_m \mid \mathcal{E}_i \in \mathcal{S}_{\mathcal{E}_i}, \mathcal{S}_{\mathcal{E}_i} \in \Sigma, m = |\Sigma|\}$
16: $\quad \mathcal{S_E} \leftarrow \{\mathcal{E} \in \mathcal{S_E} \mid \mathcal{E}$ is minimal, consistent, and relevant$\}$
17: **end if**
18: **return** $\mathcal{S_E}$

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 results is `"nothing to explain"`.

If Σ is non-empty the explanations of $\mathcal{P}$ are computed as unions of the partial explanations $\mathcal{E}_1 \cup \cdots \cup \mathcal{E}_m$, 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).

We will now show that also $\mathrm{AAA_S}$ is correct.

Lemma 4 (Soundness)**.** *Let $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ be a multiple-observation abduction problem, and let $l \geq 1$. Let $\mathcal{S_E} := AAA_S$ $(\mathcal{K}, \mathcal{O}, l)$. Then $\mathcal{S_E} \subseteq \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$.*

Proof. If $\mathrm{AAA_S}$ returned `"nothing to explain"`, Σ was empty in line 12. This can only be the case when SOA returned `"nothing to explain"` for each O_i (line 8). This means that $\mathcal{K} \models O_i$ for each O_i, and so $\mathcal{K} \models \mathcal{O}$.

In the other case the algorithm returned a set $\mathcal{S_E}$. Let $\mathcal{E} \in \mathcal{S_E}$. From lines 15–16 it is apparent that $\mathcal{E} = \mathcal{E}_1 \cup \ldots \cup \mathcal{E}_m$ where $\mathcal{E}_i \in \mathcal{S}_{\mathcal{E}_i}$ and each $\mathcal{S}_{\mathcal{E}_i} \in \Sigma$ is the set of minimal explanations for O_i returned by SOA in line 5.

From Lemma 1, $\mathcal{K} \cup \mathcal{E}_i \models O_i$ for all $\mathcal{E}_i \in \mathcal{S}_{\mathcal{E}_i}$. Observe, that Σ collects $\mathcal{S}_{\mathcal{E}_i}$ for all those O_i, for which $\mathcal{K} \not\models O_i$ (lines 8–9), hence $\mathcal{K} \cup \mathcal{E} \models \mathcal{O}$, i.e. $\mathcal{E}$ explains $\mathcal{P} = (\mathcal{K}, \mathcal{O})$. Moreover, subset minimality, consistency, and relevance of each $\mathcal{E} \in \mathcal{S}_{\mathcal{E}}$ is consecutively verified in line 16. $\mathcal{E}$ is also explanatory, as otherwise $\mathcal{K} \models \mathcal{O}$, i.e. $\mathcal{K} \models O_i$ for all i, and thus $\Sigma = \{\}$ and the algorithm would already terminate in line 13. □

Lemma 5 (Completeness)**.** *Let $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ be a multiple-observation abduction problem, $l \geq 1$. Let $\mathcal{E} \in \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$ and let $|\mathcal{E}| \leq l$. Let $\mathcal{S}_{\mathcal{E}} := AAA_S\ (\mathcal{K}, \mathcal{O}, l)$. Then $\mathcal{E} \in \mathcal{S}_{\mathcal{E}}$.*

Proof. As $\mathcal{K} \cup \mathcal{E} \models \mathcal{O} = \{O_1, \dots, O_n\}$, then also $\mathcal{K} \cup \mathcal{E} \models O_i$ and hence $\mathcal{E}$ explains each $\mathcal{P}_i = (\mathcal{K}, O_i)$. Let $\mathcal{E}_i$ be the smallest subset of $\mathcal{E}$ that explains $\mathcal{P}_i$. For some i, $\mathcal{E}_i$ may equal to $\{\}$ but only if $\mathcal{K} \models O_i$. Hereafter we disregard all such i.

Surely there is at least one $\mathcal{E}_i \neq \{\}$ as otherwise $\mathcal{K} \models \mathcal{O}$ which is not the case. If $\mathcal{E}_i \neq \{\}$ then $\mathcal{E}_i$ is a minimal explanation of $\mathcal{P}_i$, otherwise it would not be the smallest subset of $\mathcal{E}$ that explains $\mathcal{P}_i$. It is trivially explanatory, and it is also consistent and relevant w.r.t. $\mathcal{P}_i$ (because whole $\mathcal{E}$ is). Also trivially $|\mathcal{E}_i| \leq l$ due to $\mathcal{E}_i \subseteq \mathcal{E}$. In addition $\mathcal{E} = \mathcal{E}_1 \cup \dots \cup \mathcal{E}_n$ because $\mathcal{E}_1 \cup \dots \cup \mathcal{E}_n$ explains all $O_1, \dots, O_n$ hence otherwise $\mathcal{E}$ would not be minimal.

Now, the algorithm called SOA and obtained the set of explanations $\mathcal{S}_{\mathcal{E}_i}$ for each $\mathcal{P}_i$. From Lemma 2 we have that $\mathcal{S}_{\mathcal{E}_i}$ contains all minimal, consistent, relevant, and explanatory explanations of $\mathcal{P}_i$ up to the length l. We have showed above that $\mathcal{E} = \mathcal{E}_1 \cup \dots \cup \mathcal{E}_n$ where $\mathcal{E}_i$ is minimal, consistent, relevant, and explanatory explanation for $\mathcal{P}_i$ with $|\mathcal{E}_i| \leq l$. Hence for all $\mathcal{E}_i$ we have $\mathcal{E}_i \in \mathcal{S}_{\mathcal{E}_i}$, and hence in line 15 $\mathcal{E}$ was surely added into $\mathcal{S}_{\mathcal{E}}$. But since $\mathcal{E}$ is minimal, consistent, relevant, and explanatory, it was also added to $\mathcal{S}_{\mathcal{E}}$ in line 16. □

We have proved that SOA terminates, hence apparently $\mathrm{AAA_S}$ does too. Putting this and the two lemmata above together we obtain the following correctness results for the bounded and for the unbounded case.

Theorem 5. *Let $\mathcal{P} = (\mathcal{K}, \mathcal{O})$ be a multiple-observation abduction problem, and let $l \geq 1$. Then $AAA_S\ (\mathcal{K}, \mathcal{O}, l)$ always terminates, and it is sound and complete w.r.t. all $\mathcal{E} \in \mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$ s.t. $|\mathcal{E}| \leq l$.*

Corollary 4. *Given a multiple-observation abduction problem $\mathcal{P} = (\mathcal{K}, \mathcal{O})$, $AAA_S\ (\mathcal{K}, \mathcal{O}, \infty)$ always terminates, and it is sound and complete w.r.t. $\mathrm{A_nR_n^{CER,sub}}(\mathcal{P})$.*

5.3. Complexity in case of multiple-observations

The reduction employed by $\mathrm{AAA_R}$ is polynomial (in fact, linear). Similarly in case of AAA_S we call the algorithm on slightly smaller input multiple times, however this factor is diminutive compared to the size of the knowledge base hence the complexity results established for SOA directly propagate to AAA in case of both approaches.

However, an important difference between the approaches is that compared to $\mathrm{AAA_R}$, $\mathrm{AAA_S}$ searches to a larger search space in order to be complete with respect to all explanations up to the maximum length bound. This is an interesting issue and we will focus on this also in our empirical evaluation in Section 7.3.

6. Implementation

Our algorithm is implemented in Java. Knowledge base consistency is verified using the Pellet reasoner [30] (version 2.3.1). The algorithm is integrated with Pellet at the source-code level. After the knowledge base $\mathcal{K}$ is initialized and a successful consistency check is performed, the ABox corresponding to the model of $\mathcal{K}$ is obtained using the `getABox()` method. ABox encoding of the model is extracted by methods `getTypes()` and `getOutEdges()`.

All other features, including the HS-tree construction and explanation extraction and filtering are executed by our own implementation. All optimizations presented in this paper such as HS-tree pruning and model reuse are implemented. Both $\mathrm{AAA_R}$ and $\mathrm{AAA_S}$ versions are implemented and can be switched via input parameter.

It is often meaningful to forbid loops (i.e., assertions of the form $R(a, a)$) in explanations, which may significantly reduce the search space. The algorithm allows to control this by an additional input parameter.

The implementation is available at: http://dai.fmph.uniba.sk/~pukancova/aaa/ .

7. Evaluation

The goals of the evaluation was to compare the performance of the algorithm on different ontologies, to compare the computation times for different length bounds, and to compare the two versions of the multiple-observation algorithm.

7.1. Dataset and methodology

We have chosen three ontologies for the evaluation: Family ontology[1], Coffee ontology by Carlos Mendes[2], and LUBM [Lehigh University Benchmark, 13]. The parameters of the ontologies are stated in Table 2. Coffee ontology is the biggest in the number of axioms. It has approximately the same number of concepts as LUBM. On the other hand, LUBM contains more roles, but the number of axioms is much lower. Family ontology serves to compare these bigger ontologies with an ontology containing a smaller number of axioms, concepts, and roles.

Table 2

Parameters of the ontologies

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

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.

The evaluation is split into two experiments: the first one for single observations, the second for multiple observations. Apart from exceptional cases all experiments were repeated 10 times on the same input and the results were averaged. The single observation experiment was executed up to the depth 5, whilst the multiple observation experiment was executed up to the depth 3.

All experiments were executed both without loops and with loops, for the lack of space 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.

[1]Our own small ontology of family relations: http://dai.fmph.uniba.sk/~pukancova/aaa/ont/

[2]Publicly available on Github: https://gist.githubcom/cmendesce/56e1e16aee5a556a186f512eda8dabf3

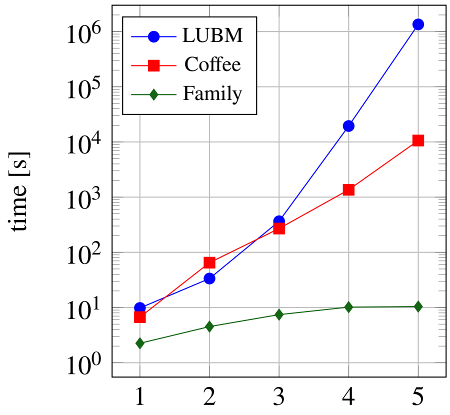

Fig. 1. Depth vs. time for single observation

7.2. Single observation experiment

In this experiment we ran AAA on each ontology with a single observation: Mother(jane) for Family ontology, Macchiato(a) for Coffee ontology, and Person(jack) for LUBM. Each run was repeated 10 times. Average execution times w.r.t. a given max HS-tree depth are plotted (in logarithmic scale) in Figure 1. The average deviation among the runs was 2.1 %.

Table 3 shows the average number of nodes in the HS-tree and the number of explanations that were found, after the HS-tree was fully traversed up to a given depth. (Note that the data are accumulative.) In Figure 2 we further analyse the generated nodes in each depth: the proportion of nodes for which TA was called, for which a model was reused, and those that were pruned is plotted here.

Note that, the sum of these nodes (i.e., 100 % in each column) amounts to the total number of nodes constructed by AAA in the given depth – i.e. the number of nodes from Table 3. These are not all nodes that would be generated if no pruning was applied at all.

We observe a significant growth of time with the growth of the depth of the HS-tree (i.e. the maximal length of explanations). This growth is much less steep in the case of the simpler Family ontology, while it is exponential with Coffee ontology and even steeper in case of LUBM. This supports the importance of the explanation length restriction. Notably, the search up to the length of 1 or 2 takes a fraction of the time required for greater lengths, and at least in our limited experiments it returned quite high number of explanations in a number of cases.

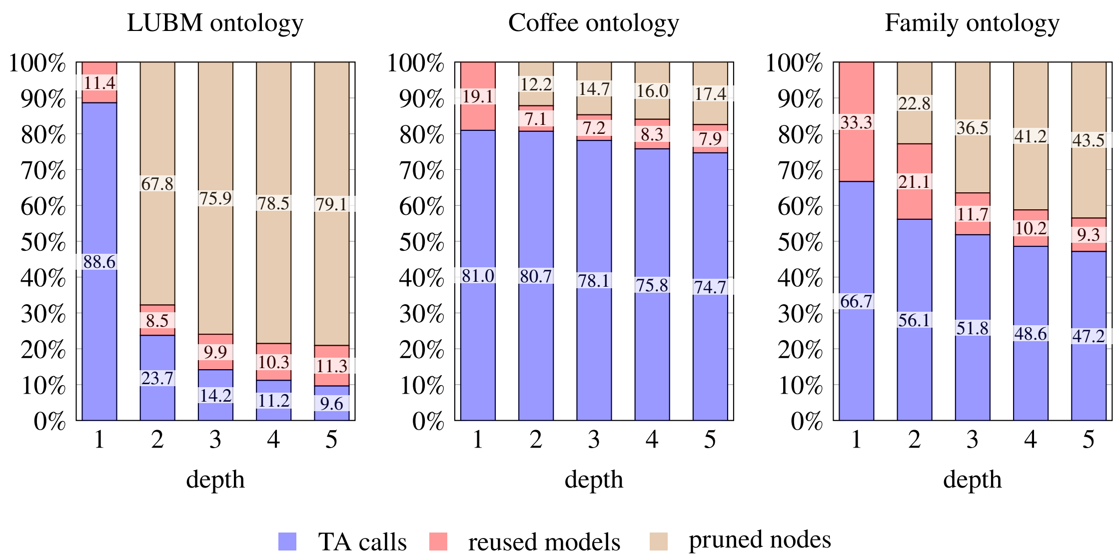

Fig. 2. Proportion of pruned nodes, reused models and TA calls for single observation

Table 3

Parameters of HS-trees for single observation

	Family		Coffee		LUBM	
Depth	**Nodes**	**Expl.**	**Nodes**	**Expl.**	**Nodes**	**Expl.**
1	9.0	1	42.0	2	44.0	20
2	57.0	4	1600.0	2	990.0	20
3	137.0	4	7627.0	2	10923.0	20
4	177.0	4	30997.0	2	84018.7	20
5	193.0	4	108519.8	2	518596.0	20

In Figure 2 we may observe that the proportion of the pruned nodes tends to rise with the increasing depth of the HS-tree. On the other hand, not so for the proportion of the reused models, which has an indifferent or even a falling trend (Family ontology). The sum of these however has an increasing trend. This result varies greatly depending on the ontology. In case of Coffee ontology it is less apparent, while in case of LUBM it is very significant. All in all, we consider the decreasing proportion of the TA calls to be a very positive result. For more expressive DLs TA is a very expensive procedure; with (potentially) exponential growth of the HS-tree, it is very important to minimize the number of TA calls as much as possible.

7.3. Multiple observation experiment

In the multiple observation experiment the algorithm was executed with the following observation sets: $\{\mathsf{Father}(\mathsf{jack}), \mathsf{Mother}(\mathsf{eva}), \mathsf{Person}(\mathsf{fred})\}$ for the Family ontology, $\{\mathsf{Milk}(\mathsf{a}), \mathsf{Coffee}(\mathsf{b}), \mathsf{Pure}(\mathsf{c})\}$ for the Coffee ontology, and $\{\mathsf{Person}(\mathsf{jack}), \mathsf{Employee}(\mathsf{jack}), \mathsf{Publication}(\mathsf{a})\}$ for LUBM. The aim was also to compare the reduction (R) and the splitting (S) approach therefore all experiments were run with either option.

Whilst the single observation experiment was processed up to the depth 5, the multiple observation experiment was processed only 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 now 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.

Analogous data were collected during this experiment, and they are shown in Table 4 and Figures 3, 4. The out-of-memory cases are missing (except for time,

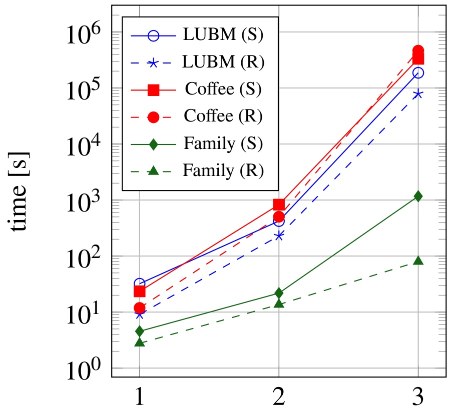

Fig. 3. Depth vs. time for multiple observations

where the value corresponds to 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.

Most of the conclusions from the single-observation case were basically confirmed. Consequently we focused on the comparison between the two approaches. The 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. This is apparent e.g. from Table 4 (Family ontology). The reduction found all 9 explanations up to length 3 after 3.9 hours. The splitting found 7 of these in 93 seconds (after depth 1) and it already found 144 explanations in 25.8 minutes (after depth 2). In fact, it took slightly longer to run it to depth 3 (4.7 hours) 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.

8. Related work

The work of Halland and Britz [15, 16] is most directly related to ours. However, Halland and Britz stay on the theoretical level, and they also compute all DL models in a preprocessing step, which is much less efficient. Similarly to other earlier works on ABox abduction [22, 23] the DL expressivity is limited (from $\mathcal{ALC}$ to $\mathcal{ALCI}$). Full soundness and completeness was only achieved by Klarman et al. [22].

Du et al. [8] provide an implementation and an interesting evaluation, however due to translation into Prolog the work is only complete up to a Horn fragment of $\mathcal{SHIQ}$.

The approach of Del-Pinto and Schmidt [6], based on forgetting, is sound and complete, and includes an implementation, but the expressivity is limited to $\mathcal{ALC}$.

In comparison, we present an approach which has no upper limit on expressivity (any reasoner can be plugged-in as a black box), adds the length limitation, it is sound and complete (up to any length), and it is implemented.

An ABox abduction service is part of RacerPro [14]. It is based on backward chaining of DL-safe rules [3, 11]. To our best knowledge, soundness and completeness results were not published in the literature.

A closely related is the more general problem of query abduction [2]. This generality comes at some cost, as noted e.g. by Du et al. [7] who provide a query-based abduction algorithm for a restricted class of TBoxes called first-order rewritable. Our approach is able to answer more specific abduction problems but with no limits on the knowledge base expressivity.

9. Conclusions

We have described ABox abduction algorithm based on MHS [28]. We have implemented this algorithm into the AAA solver which is publicly available.

Our algorithm handles multiple observations in form of any ABox assertion, and supports the class of explanations including atomic and negated atomic concept and role assertions. The algorithm aims at expressive DLs, it plugs in a DL reasoner as a black box, hence the DL expressivity is only limited by the used reasoner. Our implementation is based on Pellet [30], thus the expressivity ranges up to $\mathcal{SROIQ}$.

The algorithm always searches for the explanations starting from the shorter and it iteratively explores

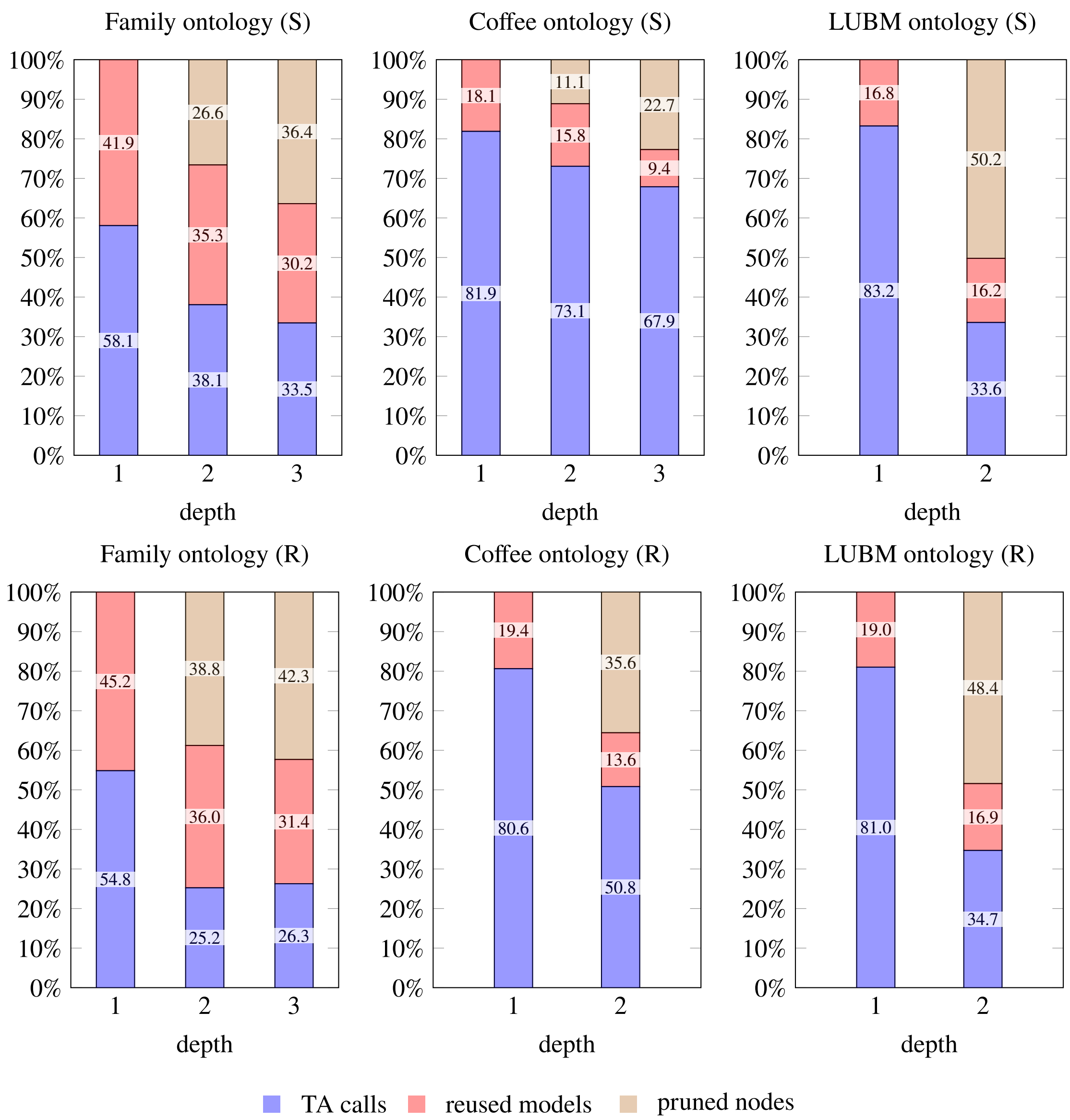

Fig. 4. Proportion of pruned nodes, reused models and TA calls for multiple observations

Table 4
Parameters of HS-trees for multiple observations

App.	Depth	Family Nodes	Family Expl.	Coffee Nodes	Coffee Expl.	LUBM Nodes	LUBM Expl.
S	1	93.0	7	480.0	12	411.0	320
	2	1545.0	144	41303.0	12	39756.3	320
	3	17072.6	813	2052996.0	12	–	–
R	1	31.0	0	160.0	0	137.0	0
	2	931.0	0	25441.0	0	18633.0	320
	3	13951.0	9	–	–	–	–

longer and longer explanation candidates. It is possible to limit this search by a maximal length l.

We have formally proven soundness and completeness of the algorithm. We have provided an exponential upper bound for the algorithm (disregarding the complexity of the DL reasoner called as a black box). This reflects the overall hardness of the minimal hitting set problem which is NP-complete [21]. Combining this with the complexity of reasoning for expressive DLs we obtain that for those DLs whose complexity is ExpTime the combined complexity is still in ExpTime, and for more complex DLs such as $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$ the combined complexity is inherited from the DL reasoner.

Searching for explanations in such a general setting, without any further restrictions is indeed computationally very expensive. However, the user may not be able to provide additional restrictions (i.e., abducibles [4]), hence the general case is also interesting.

In line with this observation, in our empirical evaluation we have focused especially on the cases with limited maximal length of explanations. Our evaluation shows that computing all explanations up to a few lower lengths is feasible. In fact, these explanations are the most preferred. We have also showed the implemented optimization techniques to be effective in reducing the search space, which we were able to study on different ontologies.

We have also empirically compared the two different versions of the multiple observation algorithm, observing that the reduction-based approach is more effective when the task is to find all explanations up to certain maximum length, while the splitting-based approach may be more preferred in cases when completeness is not of utter importance.

The AAA solver is subject to our ongoing work. In future we would like to extend our algorithm and the implementation. We would like to explore possible improvements in the MHS algorithm [e.g., 12, 32] to further boost the performance. We would also like to introduce the option to define abducibles, and we would like to plug in and compare different reasoners. Particularly the latter two extensions would also enable us to conduct a more detailed and interesting evaluation.

Acknowledgements

This work was supported from the Slovak national project VEGA 1/0778/18. We would like to thank to anonymous reviewers from DL Workshops for valuable suggestions.

References

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

[2] D. Calvanese, M. Ortiz, M. Simkus and G. Stefanoni, Reasoning about explanations for negative query answers in DL-Lite, *Journal of Artificial Intelligence Research* **48** (2013), 635–669.

[3] S. Castano, I.S. Espinosa Peraldí, A. Ferrara, V. Karkaletsis, A. Kaya, R. Möller, S. Montanelli, G. Petasis and M. Wessel, Multimedia interpretation for dynamic ontology evolution, *Journal of Logic and Computation* **19**(5) (2009), 859–897.

[4] L. Console, D.T. Dupré and P. Torasso, On the relationship between abduction and deduction, *Journal of Logic and Computation* **1**(5) (1991), 661–690.

[5] P.T. Cox and T. Pietrzykowski, Causes for events: Their computation and applications, in: *8th International Conference on Automated Deduction (CADE 1986), Oxford, England*, J.H. Siekmann, ed., LNCS, Vol. 230, Springer, 1986, pp. 608–621.

[6] W. Del-Pinto and R.A. Schmidt, Forgetting-based abduction in $\mathcal{ALC}$, in: *Workshop on Second-Order Quantifier Elimination and Related Topics (SOQE 2017), Dresden, Germany*, CEUR-WS, Vol. 2013, 2017, pp. 27–35.

[7] J. Du, K. Wang and Y. Shen, A tractable approach to ABox abduction over description logic ontologies, in: *Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, Québec, Canada.*, 2014, pp. 1034–1040.

[8] J. Du, G. Qi, Y. Shen and J.Z. Pan, Towards practical ABox abduction in large description logic ontologies, *Int. J. Semantic Web Inf. Syst.* **8**(2) (2012), 1–33.

[9] T. Eiter and G. Gottlob, The complexity of logic-based abduction, *Journal of the ACM* **42**(1) (1995), 3–42.

[10] C. Elsenbroich, O. Kutz and U. Sattler, A case for abductive reasoning over ontologies, in: *OWLED*06 Workshop on OWL: Experiences and Directions, Athens, Georgia, USA*, CEUR-WS, Vol. 216, 2006.

[11] I.S. Espinosa Peraldí, A. Kaya and R. Möller, Formalizing multimedia interpretation based on abduction over description logic ABoxes, in: *22nd International Workshop on Description Logics (DL 2009), Oxford, UK*, CEUR-WS, Vol. 477, 2009.

[12] R. Greiner, B.A. Smith and R.W. Wilkerson, A correction to the algorithm in Reiter's theory of diagnosis, *Artificial Intelligence* **41**(1) (1989), 79–88.

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

[14] V. Haarslev, K. Hidde, R. Möller and M. Wessel, The RacerPro knowledge representation and reasoning system, *Semantic Web* **3**(3) (2012), 267–277.

[15] K. Halland and K. Britz, 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*, 2012a, pp. 51–58.

[16] K. Halland and K. Britz, Naïve ABox abduction in $\mathcal{ALC}$ using a DL tableau, in: *2012 International Workshop on Description Logics, (DL2012), Rome, Italy*, CEUR-WS, Vol. 814, 2012b.

[17] I. Horrocks, The FaCT system, in: *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX'98, Oisterwijk, The Netherlands*, LNCS, Vol. 1397, Springer, 1998a, pp. 307–312.

[18] I. Horrocks, Using an expressive description logic: FaCT or fiction?, in: *Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Trento, Italy*, AAAI, 1998b, pp. 636–649.

[19] I. Horrocks, O. Kutz and U. Sattler, The even more irresistible $\mathcal{SROIQ}$, in: *Tenth International Conference on Principles of Knowledge Representation and Reasoning, Lake District of the United Kingdom*, AAAI, 2006, pp. 57–67.

[20] T. Hubauer, C. Legat and C. Seitz, 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*, AISC, Vol. 88, Springer, 2011, pp. 47–56.

[21] R.M. Karp, 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.*, 1972, pp. 85–103.

[22] S. Klarman, U. Endriss and S. Schlobach, ABox abduction in the description logic $\mathcal{ALC}$, *Journal of Automated Reasoning* **46**(1) (2011), 43–80.

[23] Y. Ma, T. Gu, B. Xu and L. Chang, 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*, IFIP AICT, Vol. 385, Springer, 2012, pp. 125–130.

[24] C.S. Peirce, Deduction, induction, and hypothesis, *Popular Science Monthly* **13** (1878), 470–482.

[25] Y. Peng and J.A. Reggia, *Abductive inference models for diagnostic problem-solving*, Springer, 1990.

[26] G. Petasis, R. Möller and V. Karkaletsis, BOEMIE: Reasoning-based information extraction, in: *1st Workshop on Natural Language Processing and Automated Reasoning, A Corunna, Spain, September 15th, 2013.*, CEUR-WS, Vol. 1044, 2013, pp. 60–75.

[27] J. Pukancová and M. Homola, Abductive reasoning with description logics: Use case in medical diagnosis, in: *28th International Workshop on Description Logics (DL 2015), Athens,Greece*, CEUR-WS, Vol. 1350, 2015.

[28] R. Reiter, A theory of diagnosis from first principles, *artificial intelligence* **32**(1) (1987), 57–95.

[29] R. Shearer, B. Motik and I. Horrocks, HermiT: A highly-efficient OWL reasoner, in: *Fifth OWLED Workshop on OWL: Experiences and Directions, Karlsruhe, Germany*, CEUR-WS, Vol. 432, 2009.

[30] E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur and Y. Katz, Pellet: A practical OWL-DL reasoner, *Journal of Web Semantics* **5**(2) (2007), 51–53.

[31] A. Steigmiller, T. Liebig and B. Glimm, Konclude: System description, *Journal of Web Semantics* **27** (2014), 78–85.

[32] F. Wotawa, A variant of Reiter's hitting-set algorithm, *Information Processing Letters* **79**(1) (2001), 45–51.