Concept Abduction for Description Logics

Birte Glimm[1], Yevgeny Kazakov[1] and Michael Welt[1]

[1]Ulm University, Germany

Abstract

We present two alternative algorithms for computing (all or some) solutions to the concept abduction problem: one algorithm is based on Reiter's hitting set tree algorithm, whereas the other one relies on a SAT encoding. In contrast to previous work, the algorithms do not rely on a refutation-based calculus and, hence, can be used also with efficient reasoners for tractable DLs such as $\mathcal{EL}$ and its extensions. An adaptation to other forms of (logic-based) abduction, e.g., to ABox abduction, is also possible.

Keywords

Description Logics, abduction, ontologies, reasoning

1. Introduction

The reasoning service of *abduction*, as originally introduced by Peirce [1], is a form of reasoning that is used to generate possible *explanations* for a given *observation*. In logic-based abduction, the objective is to find a set of *hypotheses* (the explanation) which are consistent with a background theory and which logically entail the observation when taken together with the background knowledge. Often some minimality criterion, e.g., set inclusion or cardinality, is used to select preferred explanations. Abduction has numerous applications in artificial intelligence, e.g., in diagnosis, planning, or natural language understanding and abductive reasoning has intensively been studied in the context of propositional logic (see, e.g., [2]) and also in the context of answer set programming (see, e.g., [3]). For logic-based abduction over ontologies, approaches started to emerge more recently (see, e.g., [4, 5, 6, 7]). With an exception of recent approaches based on forgetting [8, 9], practical approaches for logic-based abduction over ontologies are based on refutation-based calculi such as tableau. Such approaches are based on the observation that the knowledge base $\mathcal{K}$ does not yet entail the observation O, i.e., $\mathcal{K} \cup \{\neg O\}$ is satisfiable, whereas an explanation E has to be such that $\mathcal{K} \cup E \cup \{\neg O\}$ is unsatisfiable since it is desired that $\mathcal{K} \cup E \models O$. Hence, open branches in a tableau for $\mathcal{K} \cup \{\neg O\}$ can "guide" the process of finding explanations as explanations have to be such that they lead to closing the open branches.

While refutation-based calculi are dominantly used for reasoning in expressive Description Logics (DLs), tractable reasoning procedures for DLs such as $\mathcal{EL}$ are not based on refutation,

DL 2022: 35th International Workshop on Description Logics, August 7–10, 2022, Haifa, Israel

birte.glimm@uni-ulm.de (B. Glimm); yevgeny.kazakov@uni-ulm.de (Y. Kazakov); michael.welt@uni-ulm.de (M. Welt)

https://www.uni-ulm.de/in/ki/glimm/ (B. Glimm); https://www.uni-ulm.de/in/ki/kazakov/ (Y. Kazakov)

0000-0002-6331-4176 (B. Glimm); 0000-0002-3011-8646 (Y. Kazakov)

but directly derive (logical) consequences, which prevents the use of existing approaches to abduction. We aim at closing this gap and, inspired by the work of Kazakov and Glimm [10] for computing justification (i.e., minimal subsets of axioms from a knowledge base that are needed for a given entailment), we propose an alternative approach to efficiently compute some or all (minimal) explanations. The approach is based on Reiter's minimal hitting set algorithm [11] without relying on a refutation-based calculus to guide the algorithm. An encoding of the problem into SAT allows for better managing the combinatorial problem of selecting what constitutes an explanation. As our goal is the development of an approach for DLs such as $\mathcal{EL}$, where the main reasoning task involves determining concept subsumptions, we focus on *concept abduction* [12] as introduced next. The accompanying technical report gives further details, examples, and complete proofs [13].

2. Preliminaries

We use the standard syntax and semantics of DLs (see, e.g., [14]). Note that the DL $\mathcal{EL}$ only allows for using the top concept, conjunctions, and existential restrictions as concepts.
We now introduce the basic ideas of (concept) abduction (see also [12, 15]).

Definition 1. *A* concept abduction problem $\mathcal{P}$ *is a tuple* $\langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ *with* $\mathcal{K}$ *a knowledge base,* $\mathcal{H}$*, the* hypotheses*, a set of atomic concepts, and* O*, the* observation*, a single atomic concept.*[1]
A set $E = \{A_1, \ldots, A_n\} \subseteq \mathcal{H}$ *is an* explanation *for* $\mathcal{P}$ *if* $\mathcal{K} \models A_1 \sqcap \ldots \sqcap A_n \sqsubseteq O$*. Such an* E *is* explanatory *if* $A_1 \sqcap \ldots \sqcap A_n \sqsubseteq O \notin \mathcal{K}$*,* E *is* satisfiable *if* $A_1 \sqcap \ldots \sqcap A_n$ *is satisfiable w.r.t.* $\mathcal{K}$*,* E *is* relevant *if* $\emptyset \not\models A_1 \sqcap \ldots \sqcap A_n \sqsubseteq O$*, and* E *(syntactically)* minimal *if there is no other explanation* E' *for* $\mathcal{P}$ *such that* $E' \subsetneq E$*.*

Note that the given set of hypothesis allows for restricting the set of concepts that can be used in explanations, but may also contain all concepts used in the knowledge base.

In the remainder, we are interested in finding explanations that are explanatory, satisfiable, relevant, and (syntactically) minimal. Abusing notation, for a set of concepts $E = \{A_1, \ldots, A_n\}$, we also write $\mathcal{K} \models E \sqsubseteq O$ instead of $\mathcal{K} \models A_1 \sqcap \ldots \sqcap A_n \sqsubseteq O$.

Example 1. *Consider the knowledge base* $\mathcal{K}$ *containing the axioms*

$$A \sqsubseteq \exists r.B \quad (1) \qquad\qquad C \sqcap C' \sqsubseteq D \quad (3)$$
$$\exists r.B \sqsubseteq C \quad (2) \qquad\qquad A \sqcap B \sqsubseteq \bot \quad (4)$$

and the abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ *with* $\mathcal{H} = \{A, B, C, C'\}$ *and* $O = D$*. From Axiom* (1) *and Axiom* (2)*, we have* $\mathcal{K} \models A \sqsubseteq C$*. Hence, together with Axiom* (3)*, we find that* $\mathcal{K} \models A \sqcap C' \sqsubseteq D = O$*. Among others, we have the following explanations for* $\mathcal{P}$*:*

$$E_1 = \{A, C'\} \qquad E_2 = \{C, C'\} \qquad E_3 = \{A, B\} \qquad E_4 = \{A, C, C'\}$$

[1] Note that requiring the observation to be an atomic concept is without loss of generality since for an observation in form of a complex concept C, we can simply introduce an axiom $O \equiv C$ and use O as observation. We can proceed analogously for hypotheses.

Algorithm 1: Finding one explanation

Minimize($\mathcal{K}, \mathcal{H}, O$): compute a minimal explanation for the concept abduction problem $\langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$

input : a knowledge base $\mathcal{K}$, a set of hypotheses $\mathcal{H}$, and an observation O such that $\mathcal{K} \models \mathcal{H} \sqsubseteq O$

output : a minimal explanation $E \subseteq \mathcal{H}$ such that $\mathcal{K} \models E \sqsubseteq O$ (cf. Definition 1)

```
1 E ← H;
2 for A ∈ H do
3     if K ⊨ (E \ {A}) ⊑ O then
4         E ← E \ {A};
5 return E;
```

Among these only E_1 is explanatory, relevant, satisfiable, and minimal. We have that E_2 is not explanatory since $C \sqcap C' \sqsubseteq D \in \mathcal{K}$, and E_3 is not satisfiable (due to Axiom (4)). Finally, E_4 is not minimal since $E_1 \subsetneq E_4$. Note that $\{D\}$ is not an explanation since $D \notin \mathcal{H}$, but even if it were, the explanation would not be relevant since $\emptyset \models D \sqsubseteq D$.

3. Computing Abductive Explanations

Before proposing our approach, we point out that the number of (minimal) explanations for a given observation may be exponential in the size of the knowledge base. We start by showing how to compute one explanation before generalizing the approach to compute all explanations, first, using hitting set trees and, then, via a SAT encoding.

3.1. Computing One Abductive Explanation

Given a concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$, a naive algorithm for computing one explanation E such that $\mathcal{K} \models E \sqsubseteq O$ is relatively easy. If $\mathcal{K} \not\models \mathcal{H} \sqsubseteq O$, there is no satisfiable explanation and we can stop. Otherwise, we start from $E = \mathcal{H}$ and repeatedly remove concepts from E as long as this does not break the entailment $\mathcal{K} \models E \sqsubseteq O$. At a certain point, no concept can be removed without breaking the entailment, which implies that E is a minimal explanation w.r.t. $\mathcal{P}$. Algorithm 1 summarizes this idea. Note that Algorithm 1 makes calls to a reasoning service for concept subsumption checking without relying on a particular kind of procedure (e.g., tableau-based as well as consequence-based procedures may be used). It remains to check whether the returned E is explanatory, satisfiable, and relevant, which requires at most two further subsumption checks and a check for set containment.

The correctness of Algorithm 1 relies on the fact that a conjunction with additional conjuncts is more specific:

Lemma 1. *Let $\mathcal{K}$ be a DL knowledge base, E and E' two sets of atomic concepts such that $E' \subseteq E$, and O an atomic concept. Then $\mathcal{K} \models E' \sqsubseteq O$ implies $\mathcal{K} \models E \sqsubseteq O$.*

Proof Sketch. Intuitively, the interpretation of a conjunction is more restricted the more conjuncts are contained in the conjunction. □

Note that Lemma 1 only relies on the semantics of conjunctions and subsumption, which is shared among all DLs. We show that Algorithm 1 is correct under the assumed semantics.

Theorem 1. *Let E be the output of Algorithm 1 for the input $\mathcal{K}$, $\mathcal{H}$, and O such that $\mathcal{K} \models E \sqsubseteq O$. Then E is a minimal explanation for the abduction problem $\langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$.*

Proof Sketch. Since $\mathcal{K} \models \mathcal{H} \sqsubseteq O$ by assumption, E is initialized such that $\mathcal{K} \models E \sqsubseteq O$ (in Line 1). Since we only remove a concept from E, if the subsumption still holds in Line 4, $\mathcal{K} \models E \sqsubseteq O$ is preserved. We can further show that E is minimal explanation as unneeded conjuncts are step by step removed. □

Finally, observe that a run of Algorithm 1 requires exactly n subsumption tests, where $n = \|\mathcal{H}\|$, which is bounded by the number of concepts in $\mathcal{K}$ (two further subsumption tests and a set containment test are needed for checking whether E is relevant, satisfiable, and explanatory). Hence, the complexity of computing one minimal explanation is bounded by a linear function over the complexity of subsumption checking. In particular, for tractable languages such as $\mathcal{EL}$ and its tractable extensions [16], one (minimal) explanation for the concept abduction problem can be computed in polynomial time, which is worst-case optimal [12].[2] For DLs where subsumption checking is ExpTime-complete such as $\mathcal{ALC}$ [17, 18], the algorithm requires worst-case exponential time.

3.2. Computing All Abductive Explanations

An explanation for a concept abduction problem $\langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ consists of concepts that are subsumed by the observation. As we have seen in Example 1, there can be several different explanations. To find all explanations, it is, therefore, necessary to change a concept in every explanation of $\langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ and the question of how to compute *all* (minimal) explanations arises.

Note first, that the output of Algorithm 1 depends on the *order* in which the axioms in $\mathcal{K}$ are enumerated in the for-loop (Line 2). Different orders of the concepts can result in different removals and, consequently, different explanations.

Lemma 2. *For each (minimal) explanation E of a concept abduction problem $\langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$, there exists some order of concepts in $\mathcal{H}$ for which Algorithm 1 with the input $\mathcal{K}$, $\mathcal{H}$, and O returns E.*

Proof Sketch. Assume, to the contrary of what is to be shown, that a minimal explanation E_k is not returned. We can show that processing first concepts in $\mathcal{H} \setminus E_k$ and then concepts from E_k leads to a contradiction as desired. □

[2] Note that although subsumption checking for $\mathcal{EL}$++ is tractable, the complexity of checking whether an explanation exists is found to be NP-complete by Bienvenu [12]. This apparent contradiction is because unsatisfiable explanations (which can only occur in logics that can express unsatisfiable concepts/$\bot$, as $\mathcal{EL}$++) are excluded in the definition of Bienvenu as explanations. Our algorithm, however, might terminate with an unsatisfiable explanation. If that has to be avoided, some form of backtracking for such cases would be needed, which indeed then leads to a higher complexity of the problem.

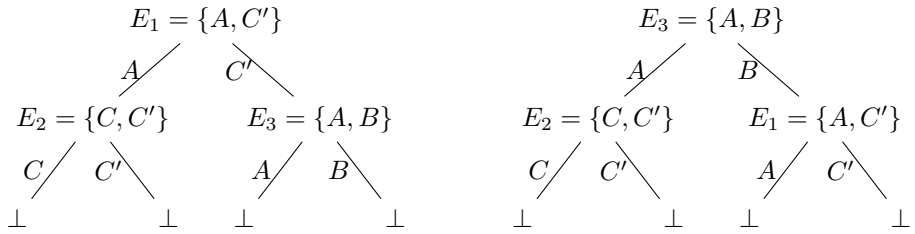

Figure 1: Two HS-trees for the concept abduction problem from Example 1

The property stated in Lemma 2 means that for computing all explanations for $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$, it is sufficient to run Algorithm 1 for all possible orders of concepts in $\mathcal{H}$. As the number of explanations can be exponential in n, the exponential behavior of an algorithm for computing *all* explanations cannot be avoided in general. Unfortunately, the described algorithm is not very practical since it performs exponentially many subsumption tests for *all* inputs, even if, e.g., $\mathcal{P}$ has just one explanation. This is because this algorithm is *not goal-directed*: the computation of each next explanation does not depend on the explanations computed before.

Hence, the question arises, how we can we find a more goal-directed algorithm. Suppose that we have computed an explanation E_1 using Algorithm 1. The next explanation E_2 must be different from E_1, so E_2 should miss at least one axiom from E_1. Hence the next explanation E_2 can be found by finding $A_1 \in E_1$ such that $\mathcal{K} \models (\mathcal{H} \setminus \{A_1\}) \sqsubseteq O$ and calling Algorithm 1 for the input $\mathcal{K}$, $\mathcal{H} \setminus \{A_1\}$, and O. The next explanation E_3, similarly, should miss something from E_1 and something from E_2, so it can be found by finding some $A_1 \in E_1$ and $A_2 \in E_2$ such that $\mathcal{K} \models (\mathcal{H} \setminus \{A_1, A_2\}) \sqsubseteq O$ and calling Algorithm 1 for the input $\mathcal{K}$, $(\mathcal{H} \setminus \{A_1, A_2\})$, and O. In general, when explanations E_i $(1 \leq i \leq k)$ are computed, the next explanation can be found by calling Algorithm 1 for the input $\mathcal{K}$, $(\mathcal{H} \setminus \{A_i \mid 1 \leq i \leq k\})$, and O such that $A_i \in E_i$ $(1 \leq i \leq k)$ and $\mathcal{K} \models (\mathcal{H} \setminus \{A_i \mid 1 \leq i \leq k\}) \sqsubseteq O$. Enumeration of subsets $\mathcal{H} \setminus \{A_i \mid 1 \leq i \leq k\}$ can be organized using a data structure called a *hitting set tree*.

Definition 2. *Let $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ be a concept abduction problem. A* hitting set tree *(short:* HS-tree*) for $\mathcal{P}$ is a labeled tree $T = (V, E, L)$ with $V \neq \emptyset$ such that:*

1. *each non-leaf node $v \in V$ is labeled with an explanation $L(v) = E$ for $\mathcal{P}$ and, for each $A \in E$, v has an outgoing edge $\langle v, w \rangle \in E$ with label $L(v, w) = A$, and*
2. *each leaf node $v \in V$ is labeled by a special symbol $L(v) = \bot$.*

For each $v \in V$, let $H(v)$ be the set of edge labels appearing on the path from v to the root node of T. Then the following properties should additionally hold:

3. *for each non-leaf node $v \in V$, we have $L(v) \cap H(v) = \emptyset$, and*
4. *for each leaf node $v \in V$, we have $\mathcal{K} \not\models (\mathcal{H} \setminus H(v)) \sqsubseteq O$.*

Figure 1 shows an example of two different HS-trees for the concept abduction problem from Example 1.

The following example shows that an explanation might occur more than once in an HS-Tree:

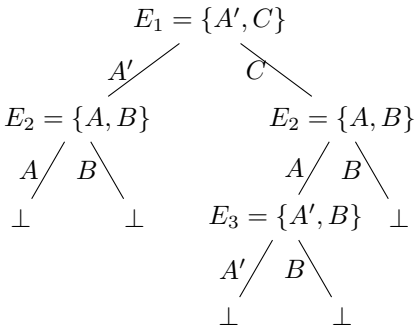

Figure 2: An HS-tree for the concept abduction problem from Example 2

Example 2. *Consider the concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ with $\mathcal{K} = \{A' \sqsubseteq A, A' \sqcap C \sqsubseteq \bot, A \sqcap B \sqsubseteq D\}$, $\mathcal{H} = \{A, A', B, C\}$, and $O = D$, which has the (minimal) explanations $E_1 = \{A', C\}$ (not satisfiable), $E_2 = \{A, B\}$ (not explanatory), and $E_3 = \{A', B\}$ (satisfiable, explanatory, and relevant). Figure 2 shows an HS-trees for $\mathcal{P}$, where the explanation $E_2 = \{A, B\}$ labels two different nodes of the tree.*

We next prove that every HS-tree must contain every explanation at least once.

Lemma 3. *Let $T = (V, E, L)$ be an HS-tree for the concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$. Then, for each explanation E for $\mathcal{P}$, there exists a node $v \in V$ such that $L(v) = E$.*

Proof Sketch. Using Lemma 1 and Conditions 1, 3 and 4 of Definition 2, we can show that a node $v \in V$ for which $H(v)$ is maximal (w.r.t. set inclusion) and $H(v) \cap E = \emptyset$ is such that $L(v) = E$. □

We next show that each HS-tree $T = (V, E, L)$ for a concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ has at most exponentially many nodes in the number of concepts in $\mathcal{H}$.

Lemma 4. *Every HS-tree T for a concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ has at most $\sum_{0 \leq k \leq n} n^k$ nodes, where n is the number of concepts in $\mathcal{H}$.*

Proof Sketch. By analyzing Conditions 1 and 3 of Definition 2, we can show that both the depth and the branching factor of T is bounded by $\|\mathcal{H}\|$, which gives the desired bound. □

For constructing an HS-tree $T = (V, E, L)$ for a concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$, we can use Reiter's *Hitting Set Tree algorithm* (or short: *HST-algorithm*) [11, 19]: We start by creating the root node $v_0 \in V$. Then we repeatedly assign labels of nodes and edges as follows. For each $v \in V$, if $L(v)$ was not yet assigned, we calculate $H(v)$. If $\mathcal{K} \not\models (\mathcal{H} \setminus H(v)) \sqsubseteq O$, we label $L(v) = \bot$ according to Condition 4 of Definition 2. Otherwise, we compute an explanation E for $\mathcal{K} \models (\mathcal{H} \setminus H(v)) \sqsubseteq O$ using Algorithm 1 and set $L(v) = E$. Note that E satisfies Condition 3 of Definition 2 since $E \subseteq (\mathcal{H} \setminus H(v))$. Next, for each $A \in E$, we create a successor node w of v and label $L(v, w) = A$. This ensures that Condition 1 of Definition 2 is satisfied for v. Since, by Lemma 4, H has a bounded number of nodes, this process eventually terminates.

Algorithm 2: Computing all explanations by the Hitting Set Tree algorithm

ComputeExplanationsHST($\mathcal{K}, \mathcal{H}, O$): compute all explanations for $\langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$
input : a knowledge base $\mathcal{K}$, a set of hypotheses $\mathcal{H}$, and an observation O such that $\mathcal{K} \models \mathcal{H} \sqsubseteq O$
output : the set of all (minimal) subsets $E \subseteq \mathcal{H}$ such that $\mathcal{K} \models E \sqsubseteq O$

```
S ← ∅;
Q ← {∅};
while Q ≠ ∅ do
    H ← choose H ∈ Q;
    Q ← Q \ {H};
    if K ⊨ H \ H ⊑ O then
        E ← Minimize(K, H \ H, O);
        S ← S ∪ {E};
        for A ∈ E do
            Q ← Q ∪ {H ∪ {A}};
return S;
```

Note that unlike the algorithm sketched in Lemma 2, the input for each call of Algorithm 1 now depends on the results returned by the previous calls.

The main idea of the HST-algorithm is to systematically compute two kinds of sets: (1) explanations E for the concept abduction problem $\langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ and (2) sets H that contain one element from each explanation E on a branch. The name of the algorithm comes from the notion of a hitting set, which characterizes the latter sets.

Definition 3. *Let P be a set of sets of some elements. A set H is a* hitting set *for P if $H \cap S \neq \emptyset$ for each $S \in P$. A hitting set H for P is* minimal *if every $H' \subsetneq H$ is not a hitting set for P.*

Intuitively, a hitting set for P is a set H that contains at least one element from every set $S \in P$. An HS-tree is then a tree $T = (V, E, L)$ such that for each $v \in V$, $H(v)$ is a hitting set of the set of explanations on the path from v to the root of T. The leaf nodes v of T are labeled by hitting sets $H(v)$ such $\mathcal{K} \not\models (\mathcal{H} \setminus H(v)) \sqsubseteq O$. Intuitively, the set $H(v)$ represents a set such that the removal of $H(v)$ from $\mathcal{H}$ breaks the subsumption.

We get following bound on the number of subsumption tests performed by the HST-algorithm:

Lemma 5. *An HS-tree for a concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ can be constructed using at most $\sum_{1 \leq k \leq n+1} n^k$ subsumption tests, where n is the number of concepts in $\mathcal{H}$.*

Proof. We call Algorithm 1 exactly once per node. Combined with Lemma 4, we get the desired bound on the number of subsumption tests performed by the HST-algorithm. □

The HST-algorithm can further be optimized in several ways. First, it is not necessary to store the complete HS-tree in memory. For computing an explanation E at each node v, it is sufficient to know just the set $H(v)$. For each successor w of v associated with some $A \in H(v)$,

the set $H(w)$ can be computed as $H(w) = H(v) \cup \{A\}$. Hence, it is possible to compute all explanations by recursively processing and creating the sets $H(v)$ as shown in Algorithm 2. The algorithm saves all explanations in a set S, which is initially empty (Line 1). The explanations are computed by processing the sets $H(v)$; the sets that are not yet processed are stored in the queue Q, which initially contains $H(v_0) = \emptyset$ for the root node v_0 (Line 2). The elements of Q are then repeatedly processed in a loop (Lines 3–10) until Q becomes empty. First, we choose any $H \in Q$ (Line 4) and remove it from Q (Line 5). Then, we test whether $\mathcal{K} \models (\mathcal{H} \setminus H) \sqsubseteq O$ (Line 6). If the subsumption holds, this means that the corresponding node v of the HS-tree with $H(v) = H$ is not a leaf node. We then compute an explanation E using Algorithm 1 and add it to S (Lines 7–8). Further, for each $A \in E$, we create the set $H(w) = H(v) \cup \{A\}$ for the corresponding successor node w of v and add $H(w)$ to Q for later processing (Lines 9–10). If the subsumption $\mathcal{K} \models (\mathcal{H} \setminus H) \sqsubseteq O$ does not hold, we have reached a leaf of the HS-tree and no further children of this node should be created.

Lemma 6. *Given $\mathcal{K}$, $\mathcal{H}$, and O as input, Algorithm 2 returns* all *explanations for the concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$.*

Proof Sketch. We can prove the claim by showing that the following invariant always holds in the main loop (Lines 3–10): If E is an explanation for $\mathcal{P}$, then either $E \in S$ or there exists $H \in Q$ such that $E \subseteq \mathcal{H} \setminus H$. □

Note that, as before, checking whether a returned subsumption is explanatory, satisfiable, and relevant requires at most two subsumption and a set membership check.

3.3. Computing Abductive Explanations using SAT Solvers

The main idea of the HST-algorithm is to systematically compute two kinds of sets: (1) explanations E for a concept abduction problem $\langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ and (2) hitting sets H that contain one element from each explanation E on a branch. Intuitively, for a leaf node v, $H(v)$ represents a set such that the removal of $H(v)$ from $\mathcal{H}$ breaks the subsumption, which means that every explanation on the path to v is also a minimal hitting set for $H(v)$. This property is called the *hitting set duality* and it takes a prominent place in the HST-algorithm. We can, however, also use this property as the basis of a *direct* algorithm for computing abductive explanations.

Suppose that we have already computed some set S of explanations and some set P of hitting sets for the concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$. How can we find a new explanation? As mentioned, each new explanation must be a hitting set for P, i.e., it should contain one concept from every set in P. Furthermore, it should be different from any of the previously computed explanations, i.e., it should miss one axiom from every $E \in S$. Suppose we have found a subset $M \subseteq \mathcal{H}$ satisfying these two requirements:

$$\forall R \in P : M \cap R \neq \emptyset, \tag{5}$$

$$\forall E \in S : E \setminus M \neq \emptyset. \tag{6}$$

If $\mathcal{K} \models M \sqsubseteq O$, then, using Algorithm 1, we can extract a *minimal* subset $E' \subseteq M$ such that $\mathcal{K} \models E' \sqsubseteq O$. Note that E' still misses at least one axiom from each $E \in S$ since (6) is preserved under removal of concepts from M. Therefore, E' is a *new explanation* for $\mathcal{P}$. If $\mathcal{K} \not\models M \sqsubseteq O$, then, similarly, by adding concepts $A \in \mathcal{H}$ to M preserving $\mathcal{K} \not\models M \sqsubseteq O$, we

Algorithm 3: Maximizing non-subsumption

Maximize($\mathcal{K}, \mathcal{H}, M, O$)**:** compute a maximal subset $N \subseteq \mathcal{H}$ such that $M \subseteq N$ and $\mathcal{K} \not\models N \sqsubseteq O$

input : a knowledge base $\mathcal{K}$, a set of hypotheses $\mathcal{H}$, a subset $M \subseteq \mathcal{H}$, and an observation O such that $\mathcal{K} \not\models M \sqsubseteq O$

output : N such that $M \subseteq N \subseteq \mathcal{H}$ such that $\mathcal{K} \not\models N \sqsubseteq O$ but $\mathcal{K} \models N' \sqsubseteq O$ for every N' with $N \subsetneq N' \subseteq \mathcal{H}$

```
N ← M;
for A ∈ H \ M do
    if K ⊭ (N ∪ {A}) ⊑ O then
        N ← N ∪ {A};
return N;
```

can find a *maximal* superset N of M ($M \subseteq N \subseteq \mathcal{H}$) such that $\mathcal{K} \not\models N \sqsubseteq O$: see Algorithm 3. Note that (5) is preserved under additions of elements to M. Thus, using any set M satisfying (5) and (6) we can find either a new explanation or a new minimal hitting set.

The question arises, how we can find a set M satisfying Conditions (5) and (6). These conditions require solving a rather complex combinatorial problem, for which propositional (SAT) solvers offer a convenient and effective way of solving such problems. In the following, we describe a propositional encoding of Conditions (5) and (6).

To formulate the propositional encoding, we assign to each concept $A \in \mathcal{H}$ a fresh propositional variable p_A. Then, every interpretation $\mathcal{I}$ determines a set $M = M(\mathcal{I}) = \{A \in \mathcal{H} \mid p_A^{\mathcal{I}} = 1\}$ of concepts whose corresponding propositional variable is true. We construct a propositional formula F such that $F^{\mathcal{I}} = 1$ if and only if $M(\mathcal{I})$ satisfies (5) and (6) for the given sets S of explanations and P of minimal hitting sets. Thus, to find a subset M satisfying (5) and (6), it is sufficient to find a model $\mathcal{I}$ of F and compute $M(\mathcal{I})$. We define F as follows:

$$F = F(S, P) = \bigwedge_{E \in S} \bigvee_{A \in E} \neg p_A \wedge \bigwedge_{R \in P} \bigvee_{A \in R} p_A. \tag{7}$$

Example 3. *Let $\mathcal{K}$ be the ontology from Example 1. We assign the propositional variables $p_A, p_B, p_C, p_{C'}$ to the concepts A, B, C, C', respectively. Let $S = \{\{A, C'\}, \{C, C'\}\}$ be the set of explanations E_1 and E_2 from Example 1 and P a set containing $\{A, C\}$, i.e., assume we have constructed the left-most branch on the HST on the left-hand side of Figure 1. Then according to (7) we have:*

$$F = F(S, P) = (\neg p_A \vee \neg p_{C'}) \wedge (\neg p_C \vee \neg p_{C'}) \wedge (p_A \vee p_C).$$

F has a model $\mathcal{I}$ with $p_A^{\mathcal{I}} = 1$ and $p_B^{\mathcal{I}} = p_C^{\mathcal{I}} = p_{C'}^{\mathcal{I}} = 0$, which gives $M(\mathcal{I}) = \{A\}$.

Once the set M determined by a model $\mathcal{I}$ of F is found, we can extract either a new explanation E or a new minimal hitting set R from M by minimizing the subsumee using Algorithm 1 or maximizing non-subsumption using Algorithm 3. After that, we can update F according to (7) and compute a new model of F, if there exist any. Once F is unsatisfiable, S contains all explanations for $\mathcal{P}$ and P contains all minimal hitting sets.

Algorithm 4: Computing all explanations using a SAT solver

ComputeExplanationsSAT($\mathcal{K}, \mathcal{H}, O$): compute all explanations for the concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$

input : a knowledge base $\mathcal{K}$, a set of hypotheses $\mathcal{H}$, and an observation O such that $\mathcal{K} \models \mathcal{H} \sqsubseteq O$

output : the set of all minimal subsets $E \subseteq \mathcal{H}$ such that $\mathcal{K} \models E \sqsubseteq O$

1 $S \leftarrow \emptyset$;
2 $F \leftarrow \top$;
3 **while** $\exists \mathcal{I} : F^{\mathcal{I}} = 1$ **do**
4 $\quad \mathcal{I} \leftarrow$ **choose** $\mathcal{I} : F^{\mathcal{I}} = 1$;
5 $\quad M \leftarrow \{A \mid p_A^{\mathcal{I}} = 1\}$;
6 $\quad$ **if** $\mathcal{K} \models M \sqsubseteq O$ **then**
7 $\quad\quad E \leftarrow \text{Minimize}(\mathcal{K}, M, O)$;
8 $\quad\quad S \leftarrow S \cup \{E\}$;
9 $\quad\quad F \leftarrow F \wedge \bigvee\{\neg p_A \mid A \in E\}$;
10 $\quad$ **else**
11 $\quad\quad N \leftarrow \text{Maximize}(\mathcal{H}, \mathcal{H}, M, O)$;
12 $\quad\quad F \leftarrow F \wedge \bigvee\{p_A \mid A \in \mathcal{H} \setminus N\}$;
13 **return** S;

Algorithm 4 summarizes the described procedure for computing all explanations for a concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ using a SAT solver. We start by creating an empty set S of explanations (Line 1) and a formula F that is always true (Line 2). Then, in a loop (Lines 3–12), as long as F is satisfiable (which is checked using a SAT solver), we take any model $\mathcal{I}$ of F (Line 4), extract the corresponding set $M = M(\mathcal{I})$ that it defines (Line 5), and check the subsumption $\mathcal{K} \models M \sqsubseteq O$. If the entailment holds, using Algorithm 1 we compute an explanation for $\mathcal{K} \models M \sqsubseteq O$ (Line 7), which, by Lemma 1, is an explanation for $\mathcal{P}$. This explanation is then added to S (Line 8) and F is extended with a new conjunct for this explanation according to (7) (Line 9). If the entailment does not hold, we compute a maximal superset N of M such that $\mathcal{K} \not\models N \sqsubseteq O$ using Algorithm 3 (Line 11) and extend F with the corresponding conjunct for the new minimal hitting set $R = \mathcal{H} \setminus N$ according to (7) (Line 12). As soon as F becomes unsatisfiable, we return the set S of computed explanations (Line 13).

Example 4. *Consider the concept abduction problem $\mathcal{P} = \langle \mathcal{K}, \mathcal{H}, \mathcal{O} \rangle$ from Example 1 and propositional encoding of concepts in $\mathcal{H}$ from Example 3. The following table shows a run of Algorithm 4 for the inputs $\mathcal{K}$, $\mathcal{H}$, and O. Every row in this table corresponds to one iteration of the while-loop (Lines 3–12). The first column gives the value of the interpretation $\mathcal{I}$ for F computed in this iteration. The second column shows the value of M computed for this interpretation and whether the entailment $\mathcal{K} \models M \sqsubseteq O$ holds. The third column shows the result of minimizing the subsumee or maximizing the non-subsumption using Algorithms 1 and 3. The last column shows the conjunct that is added to F for the corresponding explanation or minimal hitting set.*

$p_A^{\mathcal{I}}$	$p_B^{\mathcal{I}}$	$p_C^{\mathcal{I}}$	$p_{C'}^{\mathcal{I}}$	$\mathcal{K} \models^? M \sqsubseteq O$	$\mathcal{K} \models \min(M) \sqsubseteq O /$ $\mathcal{K} \not\models \max(M) \sqsubseteq O$	C
0	0	0	0	$\mathcal{K} \not\models \top \sqsubseteq O$	$\mathcal{K} \not\models \{A, C\} \sqsubseteq O$	$p_B \vee p_{C'}$
0	1	0	0	$\mathcal{K} \not\models \{B\} \sqsubseteq O$	$\mathcal{K} \not\models \{B, C\} \sqsubseteq O$	$p_A \vee p_{C'}$
1	1	0	0	$\mathcal{K} \models \{A, B\} \sqsubseteq O$	$\mathcal{K} \models \{A, B\} \sqsubseteq O$	$\neg p_A \vee \neg p_B$
1	0	0	1	$\mathcal{K} \models \{A, C'\} \sqsubseteq O$	$\mathcal{K} \models \{A, C'\} \sqsubseteq O$	$\neg p_A \vee \neg p_{C'}$
0	0	1	1	$\mathcal{K} \models \{C, C'\} \sqsubseteq O$	$\mathcal{K} \models \{C, C'\} \sqsubseteq O$	$\neg p_C \vee \neg p_{C'}$
0	1	0	1	$\mathcal{K} \not\models \{B, C'\} \sqsubseteq O$	$\mathcal{K} \not\models \{B, C'\} \sqsubseteq O$	$p_A \vee p_C$

We briefly discuss similarities and differences between Algorithm 2 and Algorithm 4: Both algorithms systematically explore subsets of $\mathcal{H}$ and minimize the subsumee from such subset to compute explanations. Algorithm 2 constructs such subsets ($\mathcal{H} \setminus H$) manually by removing one axiom appearing in the previously computed explanation (if there is any) in all possible ways. Algorithm 4 enumerates such subsets M with the help of a SAT solver. The main difference is that Algorithm 2 may encounter the same subsets many times (on different branches), whereas the propositional encoding in Algorithm 4 ensures that such subsets never repeat. Of course, an iteration of Algorithm 2 cannot be directly compared to an iteration of Algorithm 4. Both iterations use at most one call to Algorithm 1, but Algorithm 4 may also require a call to Algorithm 3, as well as checking satisfiability of F. The latter requires solving an NP-complete problem, for which no polynomial algorithm is known so far. In order to check satisfiability of F, a SAT solver usually tries several (in worst-case exponentially many) propositional interpretations until a model of F is found. As each such interpretation $\mathcal{I}$ corresponds to a subset $M(\mathcal{I}) \subseteq \mathcal{H}$, this process can be compared to the enumeration of subsets in Algorithm 2. However, a SAT solver usually implements a number of sophisticated optimizations, which make the search for models very efficient in practice, whereas the subset enumeration strategy used by Algorithm 2 is rather simplistic. Hence Algorithm 4 is likely to win in speed. On the other hand, Algorithm 4 requires saving all explanations (and minimal hitting sets) in the propositional formula F, which might result in a formula of exponential size, if the number of such explanations or hitting sets is exponential. In this regard, Algorithm 2 could be more memory efficient since saving (all) explanations is optional (see the discussion at the end of Section 3.2). Hence both algorithms have their own advantages and disadvantages.

4. Conclusions

We have presented two alternative algorithms for computing (all or some) solutions to the concept abduction problem: one algorithm is based on Reiter's hitting set tree algorithm, whereas the other one relies on a SAT encoding. It remains to be analyzed how these algorithms behave in practice and how these algorithms differ on different real-world ontologies, where an important aspect is also finding efficient *incremental* SAT solvers.

In contrast to previous work, the algorithms do not rely on a refutation-based calculus and, hence, can be used also with efficient reasoners for tractable DLs such as $\mathcal{EL}$ and its extensions. Another direction for future work is extending the approach to other forms of (logic-based) abduction, e.g., to ABox abduction and a comparison with other existing approaches, which are mostly focusing on ABox abduction.

References

[1] C. S. Peirce, Deduction, Induction, and Hypothesis, Popular Science Monthly 13 (1878) 470–482.

[2] T. Eiter, G. Gottlob, The complexity of logic-based abduction, J. ACM 42 (1995) 3–42. URL: https://doi.org/10.1145/200836.200838.

[3] M. Denecker, A. C. Kakas, Abduction in logic programming, in: A. C. Kakas, F. Sadri (Eds.), Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski, Part I, volume 2407 of *Lecture Notes in Computer Science*, Springer, 2002, pp. 402–436. URL: https://doi.org/10.1007/3-540-45628-7_16.

[4] C. Elsenbroich, O. Kutz, U. Sattler, A case for abductive reasoning over ontologies, in: B. C. Grau, P. Hitzler, C. Shankey, E. Wallace (Eds.), Proceedings of the OWLED*06 Workshop on OWL: Experiences and Directions, Athens, Georgia, USA, November 10-11, 2006, volume 216 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2006. URL: http://ceur-ws.org/Vol-216/submission_25.pdf.

[5] S. Klarman, U. Endriss, S. Schlobach, Abox abduction in the description logic *ALC*, J. Autom. Reason. 46 (2011) 43–80. URL: https://doi.org/10.1007/s10817-010-9168-z.

[6] K. Halland, A. Britz, S. Klarman, Tbox abduction in ALC using a DL tableau, in: M. Bienvenu, M. Ortiz, R. Rosati, M. Simkus (Eds.), Informal Proceedings of the 27th International Workshop on Description Logics, Vienna, Austria, July 17-20, 2014, volume 1193 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2014, pp. 556–566. URL: http://ceur-ws.org/Vol-1193/paper_42.pdf.

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

[8] P. Koopmann, W. Del-Pinto, S. Tourret, R. A. Schmidt, Signature-based abduction for expressive description logics, in: D. Calvanese, E. Erdem, M. Thielscher (Eds.), Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020, 2020, pp. 592–602. URL: https://doi.org/10.24963/kr.2020/59.

[9] W. Del-Pinto, R. A. Schmidt, Abox abduction via forgetting in ALC, in: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019, AAAI Press, 2019, pp. 2768–2775. URL: https://doi.org/10.1609/aaai.v33i01.33012768.

[10] B. Glimm, Y. Kazakov, Classical algorithms for reasoning and explanation in description logics, in: M. Krötzsch, D. Stepanova (Eds.), Reasoning Web. Explainable Artificial Intelligence - 15th International Summer School 2019, Bolzano, Italy, September 20-24, 2019, Tutorial Lectures, volume 11810 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 1–64. URL: https://doi.org/10.1007/978-3-030-31423-1_1.

[11] R. Reiter, A theory of diagnosis from first principles, Artificial Intelligence 32 (1987) 57–95. URL: https://www.sciencedirect.com/science/article/pii/0004370287900622. doi:https://doi.org/10.1016/0004-3702(87)90062-2.
[12] M. Bienvenu, Complexity of abduction in the EL family of lightweight description logics, in: G. Brewka, J. Lang (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, September 16-19, 2008, AAAI Press, 2008, pp. 220–230. URL: http://www.aaai.org/Library/KR/2008/kr08-022.php.
[13] M. W. Birte Glimm, Yevgeny Kazakov, Concept Abduction for Description Logics – Technical Report, Technical Report, Ulm University, Institute of Artificial Intelligence, 2022. https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.090/Publikationen/2022/GlKW22a_report.pdf.
[14] M. Krötzsch, F. Simančík, I. Horrocks, A description logic primer, CoRR abs/1201.4089 (2012). Available at http://arxiv.org/pdf/1201.4089.pdf.
[15] T. Eiter, G. Gottlob, The complexity of logic-based abduction, J. ACM 42 (1995) 3–42. URL: https://doi.org/10.1145/200836.200838. doi:10.1145/200836.200838.
[16] F. Baader, S. Brandt, C. Lutz, Pushing the $\mathcal{EL}$ envelope further, in: Proc. 5th Workshop on OWL: Experiences and Directions (OWLED'08), volume 496, CEUR, 2008.
[17] K. Schild, A correspondence theory for terminological logics: Preliminary report, in: J. Mylopoulos, R. Reiter (Eds.), Proc. 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91), Morgan Kaufmann, 1991, pp. 466–471.
[18] F. M. Donini, F. Massacci, EXPTIME tableaux for $\mathcal{ALC}$, J. of Artificial Intelligence 124 (2000) 87–138.
[19] R. Greiner, B. A. Smith, R. W. Wilkerson, Readings in model-based diagnosis, Morgan Kaufmann Publishers Inc., 1992, pp. 49–53.