Pinpointing Axioms in Ontologies via ASP*

Rafael Peñaloza[1][0000−0002−2693−5790] and Francesco Ricca[2][0000−0001−8218−3178]

[1] University of Milano-Bicocca, Milano, Italy
rafael.penaloza@unimib.it,
[2] University of Calabria, Rende, Italy
francesco.ricca@unical.it

Abstract. Axiom pinpointing is the task of identifying the axiomatic causes for a consequence to follow from an ontology. Different approaches have been proposed in the literature for finding one or all the subset-minimal subontologies that preserve a description logic consequence. We propose an approach that leverages the capabilities of answer set programming for transparent axiom pinpointing. We show how other associated tasks can be modelled without much additional effort.

Keywords: axiom-pinpointing, non-standard reasoning, ASP

1 Introduction

Axiom pinpointing [16] is the task of identifying the axioms in an ontology that are responsible for a consequence to follow. It has been extensively studied in description logics (DLs) and, under different names, in other areas [11, 13]. To-date, the most successful approach to axiom pinpointing which does not rely on repeated (black-box) calls to a reasoner is a reduction to MUS enumeration on a propositional formula [1, 17]. The main disadvantage of this approach is that it requires, as a pre-processing step, the construction of a huge formula, which makes the reasoning steps explicit. It is also limited to enumerating one or all so-called justifications.

We propose a novel approach based on a translation to Answer Set Programming (ASP) [7, 12]. The approach is general, and can be applied to any ontology language which allows a "modular" ASP representation in the sense that each axiom is translatable to a set of rules. We instantiate it to deal with the simple DL $\mathcal{HL}$ and the more expressive $\mathcal{EL}$. In addition to finding one or all justifications, we show that justifications of minimal cardinality and the intersection of all justifications can be easily computed through standard ASP constructs and reasoning tasks.

2 Preliminaries

We assume that the reader is familiar with the basic terminology and structure of answer set programming (ASP) [5, 7]. Here, we recall the basic ideas of de-

* This work was partially supported by MUR under PRIN project PINPOINT Prot. 2020FNEB27, CUP H23C22000280006 and H45E21000210001.

scription logics (DLs) [3], with a particular focus on the lightweight DL $\mathcal{EL}$ [2], and of axiom pinpointing [15].

Description Logics. Description logics (DLs) are a family of knowledge representation formalisms characterised by a clear syntax and a formal unambiguous semantics based on first-order logic. The main building blocks of all DLs are *concepts* (corresponding to unary predicates) and *roles* (binary predicates). The knowledge of an application domain is encoded in an *ontology*, which restricts the class of relevant interpretations of the terms, thus encoding relationships between them. Among the many existing DLs, a prominent example is the lightweight DL $\mathcal{EL}$. $\mathcal{EL}$ has a very limited expressivity, but allows for efficient (standard) reasoning tasks. For the scope of this paper, we use $\mathcal{EL}$ as a prototypical example, following the fact that most work on axiom pinpointing has focused on this logic as well. Other DLs are characterised by a different notion of concepts and a larger class of axioms.

Definition 1 ($\mathcal{EL}$). *Let N_C and N_R be two disjoint sets of* concept names *and* role names*, respectively.* $\mathcal{EL}$-concepts *are built through the grammar rule*

$$C ::= A \mid \top \mid C \sqcap C \mid \exists r.C,$$

where $A \in N_C, r \in N_R$, and $\top$ is a distinguished top concept*.*

An interpretation *is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where $\Delta^{\mathcal{I}}$ is a non-empty set called the* domain *and $\cdot^{\mathcal{I}}$ is the* interpretation function *which maps every $A \in N_C$ to a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and every $r \in N_R$ to a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. This interpretation is extended to $\mathcal{EL}$-concepts setting $\top^{\mathcal{I}} := \Delta^{\mathcal{I}}$, $(C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}$, and $(\exists r.C)^{\mathcal{I}} := \{\delta \mid \exists \eta \in C^{\mathcal{I}}.(\delta, \eta) \in r^{\mathcal{I}}\}$.*

Ontologies are finite sets of *general concept inclusions* (GCIs), which specify the relationships between concepts.

Definition 2 (ontology). *A* GCI *is an expression of the form $C \sqsubseteq D$ where C, D are two concepts. An* ontology *is a finite set of GCIs. The interpretation $\mathcal{I}$* satisfies *the GCI α iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. It is a* model *of the ontology $\mathcal{O}$ iff it satisfies all GCIs in $\mathcal{O}$. We often call GCIs* axioms*.*

The ontology $\mathcal{O}$ entails *the GCI α ($\mathcal{O} \models \alpha$) iff every model of $\mathcal{O}$ satisfies α. In this case we say that α is a* consequence *of $\mathcal{O}$.*

Although many reasoning tasks can be considered, along with an ample selection of axioms in the ontologies, we focus on the problem of deciding whether α is a consequence of an ontology. For simplicity, we will consider only *atomic* subsumption relations $A \sqsubseteq B$ where $A, B \in N_C$. It is well known that this problem can be solved in polynomial time through a completion algorithm [2]. In a nutshell, the algorithm runs in two phases. First, the original GCIs are decomposed into a set of GCIs in *normal form*; that is, having only the shapes

$$A_1 \sqsubseteq B, \quad A_1 \sqcap A_2 \sqsubseteq B, \quad A_1 \sqsubseteq \exists r.B, \quad \exists r.A_1 \sqsubseteq B$$

where $r \in N_R$ and $A, B \in N_C \cup \{\top\}$. These axioms are then combined through *completion rules* to make consequences explicit (more details in Section 3). The method is sound and complete for all atomic subsumptions over the concept names appearing in the original ontology.

As an additional example of a logic, we consider the sublanguage $\mathcal{HL}$ of $\mathcal{EL}$, which uses only concept names and the conjunction ($\sqcap$) constructor. It can be seen that $\mathcal{HL}$ is a syntactic variant of directed hypergraphs. Specifically, a GCI $A_1 \sqcap \cdots \sqcap A_m \sqsubseteq B_1 \sqcap \cdots \sqcap B_n$ represents a directed hypergraph connecting nodes $A_1, \ldots, A_m$ with nodes $B_1, \ldots, B_n$, and the entailment problem is nothing more than reachability in this hypergraph.

Axiom Pinpointing. Beyond standard reasoning, it is sometimes important to understand which axioms are responsible for a consequence to follow from an ontology. This goal is interpreted as the task of identifying *justifications*.

Definition 3. *A* justification *for a consequence α w.r.t. the ontology $\mathcal{O}$ is a set $\mathcal{M} \subseteq \mathcal{O}$ such that (i) $\mathcal{M} \models \alpha$ and (ii) for every $\mathcal{N} \subset \mathcal{M}$, $\mathcal{N} \not\models \alpha$.*

In words, a justification is a subset-minimal subontology that still entails the consequence. Most work focuses on computing one or all justifications. While the former problem remains polynomial in $\mathcal{EL}$, the latter necessarily needs exponential time, as the number of justifications may be exponential on the size of the ontology. Despite some potential uses, which have been identified for non standard reasoning [6], only very recently have specific algorithms for computing the unions and intersection of justifications been developed [9, 14]. To the best of our knowledge, no previous work has considered computing the justifications of *minimal cardinality* directly.

3 Reasoning Through Rules

Before presenting our approach to axiom pinpointing using ASP, we briefly describe how to reduce reasoning in $\mathcal{EL}$ to ASP. The approach simulates the completion algorithm sketched in Section 2 through a small set of rules, while the ontology axioms (in normal form) are represented through facts.

Consider an ontology $\mathcal{O}$ in normal form, and let $\mathcal{C}(\mathcal{O})$ and $\mathcal{R}(\mathcal{O})$ be the sets of concept names and role names appearing in $\mathcal{O}$, respectively. For each $A \in \mathcal{C}(\mathcal{O})$ we use a constant `a`, and for each $r \in \mathcal{R}(\mathcal{O})$ we use a constant `r`. We identify the four shapes of normal form axioms via a predicate. Hence, `s1(a,b)` stands for the GCI $A \sqsubseteq B$ and analogously for the expressions `s2(a1,a2,b)`, `s3(a,r,b)`, and `s4(r,a,b)`. For each axiom in normal form appearing in $\mathcal{O}$, we write the associated fact. As previously mentioned, the reasoning process is simulated through rules. In the specific case of $\mathcal{EL}$, these rules are shown in Figure 1 (left). To decide whether the atomic subsumption $A \sqsubseteq B$ is a consequence of the ontology, we need only ask the query `s1(a,b)`. Since the original ontology may not be in normal form, the facts obtained this way are the result of the normalisation step over the original GCIs. In the case of $\mathcal{HL}$, one can produce a

```
s1(X,Y) :- s1(X,Z), s1(Z,Y).
s1(X,Y) :- s1(X,Z1), s1(X,Z2), s2(Z1,Z2,Y).
s3(X,R,Y) :- s1(X,Z), s3(Z,R,W), s1(W,Y).
s1(X,Y) :- s3(X,R,Z), s4(R,Z,Y).
```

```
gi :- a1, ..., am
b1 :- gi
⋮
bn :- gi
```

Fig. 1. The rules for $\mathcal{EL}$ reasoning (left) and the translation of $\mathcal{HL}$ GCIs (right).

more direct reduction, which takes into account the hyperedges without the need for normalisation or general derivation rules. We again represent each concept name A through a constant `a`, and associate a new constant `gi` for each GCI in $\mathcal{O}$. Then the GCI $A_1 \sqcap \cdots \sqcap A_m \sqsubseteq B_1 \sqcap \cdots \sqcap B_n$ is translated to the set of rules in Figure 1 (right). To decide whether $A \sqsubseteq B$ is a consequence, we add the fact `a.` and verify the query `b`. The correctness of the approach follows from the results in [8, 15].

4 Axiom Pinpointing Through ASP

We present a general approach for solving axiom pinpointing tasks through an ASP solver. The approach is applicable to any logic (including other DLs) with a *modular* ASP encoding. Roughly, an encoding is modular if each axiom in $\mathcal{O}$ translates to a set of rules, such that an ASP encoding $\Pi_\mathcal{O}$ of $\mathcal{O}$ is obtained by the union of the encodings of its axioms, possibly together with some additional rules (independent of the axioms in $\mathcal{O}$) needed to simulate reasoning in ASP.

Definition 4. *An encoding in ASP $\Pi_\mathcal{O}$ of the ontology $\mathcal{O}$ is* modular *iff (i) for each $\alpha \in \mathcal{O}$ there is an ASP program Π_α, and (ii) there is a (possibly empty) set of rules R such that $\Pi_\mathcal{O} = \bigcup_{\alpha\in\mathcal{O}} \Pi_\alpha \cup R$*

The encodings from Section 3 for $\mathcal{EL}$ and $\mathcal{HL}$ are both modular. In the former case, R is exactly the set of rules in Figure 1 (left), while in the latter $R = \emptyset$.

We now formulate the problem of computing justifications in ASP. First, we apply an *adornment* step, which allows to identify and keep track of the rules of a module corresponding to a given axiom.

Definition 5. *Let P be an ASP program, and δ be an atom not occurring in P. The δ-adornment for P is the program $\Delta(P) = \{r_\delta : r \in P\}$, where r_δ is s.t. $head(r_\delta) = head(r)$, and $body(r_\delta) = body(r) \cup \delta$.*

In words, the δ-adornment adds a new identifying atom δ to the body of each rule of the program. This guarantees that the rules *trigger* only when δ is true.

Definition 6. *The adorned ASP encoding of the ontology $\mathcal{O}$ is the program*

$$\delta(\Pi_\mathcal{O}) = \bigcup_{\alpha\in\mathcal{O}} \Delta_\alpha(\Pi_\alpha) \cup R \cup C$$

where for each $\alpha \in \mathcal{O}$, δ_α is a fresh atom not occurring in $\Pi(\mathcal{O})$, and C is the ASP program containing a choice rule $\{\delta_\alpha\}$ for each $\alpha \in \mathcal{O}$.

In the case of $\mathcal{EL}$, the adornment will change each fact (corresponding to a GCI in normal form) `si(...).` into the rule `si(...) :- xj`, where `xj` is the chosen constant for the original axiom α_j. Importantly, this approach handles the original axioms in the ontology, and not those already normalised as done e.g. in [4].

We now describe an ASP program that can be used for axiom pinpointing. Given the ontology $\mathcal{O}$ and consequence c, we identify the justifications for c through the following property.

Proposition 1. *Let $\mathcal{O}$ be an ontology, c an atom modelling a consequence of $\mathcal{O}$, and P the program $P = \delta(\Pi_{\mathcal{O}}) \cup \{\leftarrow \mathit{not}\ c\}$. $\mathcal{M} \subseteq \mathcal{O}$ is a justification for c iff there is an answer set A of P that is minimal w.r.t. $\{\delta_\alpha \mid \alpha \in \mathcal{O}\}$ and $\{\delta_\alpha \mid \alpha \in \mathcal{M}\} \subseteq A$.*

Justifications that are cardinality minimal (and thus also subset minimal) can be directly computed using an ASP program with weak constraints.

Proposition 2. *Let $\mathcal{O}$ be an ontology, c an atom modelling a consequence of $\mathcal{O}$, and P the program $P = \delta(\Pi_{\mathcal{O}}) \cup \{:-\ \mathit{not}\ c\} \cup \{:\sim \delta_\alpha : \alpha \in \mathcal{O}\}$. $\mathcal{M} \subseteq \mathcal{O}$ is a justification for c iff there exists an optimal answer set A of P such that $\{\delta_\alpha \mid \alpha \in \mathcal{M}\} \subseteq A$.*

Before concluding, we note that the translation permits computing the *intersection* of all justifications, and consequences derived from it, through the application of *cautious reasoning* [5]. In ASP, a cautious consequence is one that holds in every answer set. Since the program P from Proposition 1 provides a one-to-one correspondence between answer sets and sub-ontologies deriving a consequence, cautious reasoning refers to reasoning over the intersection of all those sub-ontologies, and in particular over the subset-minimal ones; that is, over the justifications. Unfortunately, an analogous result does not exist for the *union* of all justifications. Indeed, every axiom would be available for *brave reasoning* (consequences which hold in at least one answer set) [5] over the same program P, but not all axioms belong to some justification.

5 Conclusions

We presented a general approach for axiom pinpointing based on a reduction to ASP. As a proof of concept, we have shown how the reduction works for the light-weight DL $\mathcal{HL}$ and the more expressive $\mathcal{EL}$. The same approach works for any logic with a modular translation to ASP, for instance any DL with a consequence-based reasoning algorithm [10, 18] should enjoy such a translation. Compared to existing approaches [1,17], ours is more general and does not require the construction of a specific propositional formula encoding the reasoning task.

In future work we will extend the translation to $\mathcal{ALC}$ and more expressive DLs, and test the efficiency of our method on ASP solvers. We will also study the implementation of other axiom pinpointing services based on ASP constructs.

References

1. Arif, M.F., Mencía, C., Ignatiev, A., Manthey, N., Peñaloza, R., Marques-Silva, J.: BEACON: an efficient SAT-based tool for debugging $EL+$ ontologies. In: Proceedings of SAT 2016. LNCS, vol. 9710, pp. 521–530. Springer (2016)
2. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Proceedings of IJCAI'05. pp. 364–369. Professional Book Center (2005)
3. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, second edn. (2007)
4. Baader, F., Peñaloza, R., Suntisrivaraporn, B.: Pinpointing in the description logic EL+. In: Proc. of KI'07. LNCS, vol. 4667, pp. 52–67. Springer (2007)
5. Baral, C.: Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press (2010)
6. Bienvenu, M., Rosati, R.: Tractable approximations of consistent query answering for robust ontology-based data access. In: Rossi, F. (ed.) Proceedings of IJCAI'13. pp. 775–781. AAAI Press/IJCAI (2013)
7. Brewka, G., Eiter, T., Truszczynski, M.: Answer set programming at a glance. Commun. ACM 54(12), 92–103 (2011)
8. Ceylan, İ.İ., Mendez, J., Peñaloza, R.: The bayesian ontology reasoner is born! In: Proceedings of (ORE-2015). CEUR Workshop Proceedings, vol. 1387, pp. 8–14. CEUR-WS.org (2015)
9. Chen, J., Ma, Y., Peñaloza, R., Yang, H.: Union and intersection of all justifications. In: Proc. of ESWC 2022. LNCS, vol. 13261, pp. 56–73. Springer (2022)
10. Cucala, D.T., Grau, B.C., Horrocks, I.: Consequence-based reasoning for description logics with disjunction, inverse roles, and nominals. In: Proceedings of DL 2017, Montpellier, France, July 18-21, 2017. (2017)
11. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.A.: Debugging unsatisfiable classes in OWL ontologies. Journal of Web Semantics 3(4), 268–293 (2005)
12. Lifschitz, V.: Answer set planning. In: Schreye, D.D. (ed.) Logic Programming: The 1999 International Conference. pp. 23–37. MIT Press (1999)
13. Peñaloza Nyssen, R.: Axiom pinpointing in description logics and beyond. Ph.D. thesis, Technische Universität Dresden, Germany (2009), http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-24743
14. Peñaloza, R., Mencía, C., Ignatiev, A., Marques-Silva, J.: Lean kernels in description logics. In: Proceeding of the 14th Semantic Web Conference (ESWC 2017). LNCS, vol. 10249, pp. 518–533 (2017)
15. Peñaloza, R., Sertkaya, B.: Understanding the complexity of axiom pinpointing in lightweight description logics. Artif. Intell. 250, 80–104 (2017)
16. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Proceedings of IJCAI'03. pp. 355–360. Morgan Kaufmann Publishers Inc. (2003)
17. Sebastiani, R., Vescovi, M.: Axiom pinpointing in large EL+ ontologies via SAT and SMT techniques. Disi technical report, University of Trento (2015)
18. Simančík, F., Kazakov, Y., Horrocks, I.: Consequence-based reasoning beyond Horn ontologies. In: Proc. IJCAI'11. pp. 1093–1098. AAAI Press (2011)