Using Tableaux and Automata for Pinpointing in $\mathcal{EL}$

Rafael Peñaloza

Theoretical Computer Science TU Dresden, Germany
penaloza@tcs.inf.tu-dresden.de

1 Introduction

Axiom-pinpointing refers to the task of understanding the specific axioms that cause a consequence to follow from an ontology. The main problems related to axiom-pinpointing are the computation of minimal subontologies from which the consequence still follows (MinAs) or maximal subontologies not satisfying a property (MaNAs). Equivalently, one can try to compute a pinpointing formula, which is an encoding of all MinAs and MaNAs through a monotone Boolean formula. There has been a recent interest in pinpointing Description Logic (DL) ontologies, and in particular those using the fairly inexpressive language $\mathcal{EL}$.

One approach towards axiom-pinpointing is the so-called glass-box method, that consists in modifying an existing decision procedure so that it outputs a pinpointing formula instead of a simple yes/no answer. Recent studies have shown how to modify tableau-based [4] and automata-based [2] decision procedures into pinpointing algorithms.

The subsumption algorithm for $\mathcal{EL}$ [6] is not typically considered to be tableau-based. However, it has several characteristics that allow it to be considered an instance of so-called general tableaux [5]. Perhaps more surprising is the fact that the same algorithm can be seen as an automata-based method. In this paper we show how these two points of view influence the pinpointing extension of the algorithm, and in particular an exponential-time discrepancy in the execution time of both pinpointing approaches.

2 Pinpointing in the Description Logic $\mathcal{EL}$

In DLs, *concept descriptions* are inductively built through the application of a set of *constructors* starting with a set $\mathsf{N_C}$ of *concept names* and a set $\mathsf{R_C}$ of *role names.* Concept descriptions in the DL $\mathcal{EL}$ are formed using the three constructors appearing at the upper part of Table 1. An $\mathcal{EL}$ *ontology* or *TBox* is a finite set of *general concept inclusion axioms* (GCIs), whose syntax is shown in the last line of Table 1.

The semantics of $\mathcal{EL}$ is defined in terms of *interpretations* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where the *domain* $\Delta^{\mathcal{I}}$ is a non-empty set and the *interpretation function* $\cdot^{\mathcal{I}}$ maps each concept name $A \in \mathsf{N_C}$ to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$. This interpretation function can be extended to arbitrary concept descriptions inductively, in correspondence to the

Name	Syntax	Semantics
top	$\top$	$\Delta^{\mathcal{I}}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
exists restriction	$\exists r.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}} : (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
GCI	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

Table 1. Syntax and semantics of $\mathcal{EL}$.

semantics column of Table 1. An interpretation $\mathcal{I}$ is called a *model* of a TBox $\mathcal{T}$ if for each GCI $C \sqsubseteq D$ in $\mathcal{T}$ it holds that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.

The main decision problem in $\mathcal{EL}$ is the *subsumption problem*, which consists in deciding, for concept names $A_0, B_0 \in \mathsf{N_C}$, whether A_0 is *subsumed* by B_0 w.r.t. a TBox $\mathcal{T}$; i. e. whether for every model $\mathcal{I}$ of $\mathcal{T}$ it holds that $A_0^{\mathcal{I}} \subseteq B_0^{\mathcal{I}}$. In this case, we denote it as $A_0 \sqsubseteq_{\mathcal{T}} B_0$. In pinpointing, we are not interested in merely deciding a subsumption relation, but rather in understanding the reasons why the relation holds. We do this through the computation of a *pinpointing formula*. Intuitively, a pinpointing formula is an encoding of all sub-TBoxes of $\mathcal{T}$ from which the subsumption relation still follows; thus, it can be used to identify the associations between axioms responsible for the subsumption of concept names. To formally define this formula, we assume that every axiom $t \in \mathcal{T}$ is labeled with a unique propositional variable $\mathsf{lab}(t)$, and denote as $\mathsf{lab}(\mathcal{T})$ the set of all propositional variables labeling an axiom in $\mathcal{T}$. We identify a propositional *valuation* with the set of propositional variables that it makes true, and for a valuation $\mathcal{V} \subseteq \mathsf{lab}(\mathcal{T})$ we set $\mathcal{T}_{\mathcal{V}} = \{t \in \mathcal{T} \mid \mathsf{lab}(t) \in \mathcal{V}\}$.

Definition 1 (Pinpointing formula). *Given an $\mathcal{EL}$ TBox $\mathcal{T}$ and concept names A_0, B_0 occurring in $\mathcal{T}$, the monotone Boolean formula ϕ over $\mathsf{lab}(\mathcal{T})$ is a* pinpointing formula *for $\mathcal{T}$ w.r.t. $A_0 \sqsubseteq B_0$ if, for every valuation $\mathcal{V} \subseteq \mathsf{lab}(\mathcal{T})$ it holds: $A_0 \sqsubseteq_{\mathcal{T}_{\mathcal{V}}} B_0$ iff $\mathcal{V}$ satisfies ϕ.*

In the following sections we show two approaches for the computation of the pinpointing formula, based on different views of the subsumption algorithm presented in [6].

3 Tableau-based Pinpointing

In [5] it was shown that the subsumption algorithm for $\mathcal{EL}$ is an instance of general tableaux, and hence a pinpointing extension can be used to compute the pinpointing formula [4]. We now briefly describe this pinpointing extension and analyze its execution time.

For describing the algorithm, we first assume that the TBox is in *normal form*, where each GCI has one of the following forms: $A_1 \sqcap A_2 \sqsubseteq B$, $A \sqsubseteq \exists r.B$,

If $(A_1 \sqcap A_2 \sqsubseteq B)^p \in \mathcal{T}$	**and** $\{(X, A_1)^{\phi_1}, (X, A_2)^{\phi_2}\} \subseteq \mathcal{A}$	**then** insert $(X, B)^{\psi}$ to $\mathcal{A}$
If $(A \sqsubseteq \exists r.B)^p \in \mathcal{T}$	**and** $(X, A)^{\phi} \in \mathcal{A}$	**then** insert $(X, r, B)^{\psi}$ to $\mathcal{A}$
If $(\exists r.A \sqsubseteq B)^p \in \mathcal{T}$	**and** $\{(X, r, Y)^{\phi_1}, (Y, A)^{\phi_2}\} \subseteq \mathcal{A}$	**then** insert $(X, B)^{\psi}$ to $\mathcal{A}$

Table 2. Completion rules for subsumption in $\mathcal{EL}$.

or $\exists r.A \sqsubseteq B$, where $r \in \mathsf{R_C}$ and $A_1, A_2, A, B \in \mathsf{N_C} \cup \{\top\}$. Any TBox can be transformed to normal form in polynomial time [1].

Recall that for the definition of the pinpointing formula we assumed that each axiom $t \in \mathcal{T}$ is labeled with a propositional variable $\mathsf{lab}(t)$. The pinpointing extension of the subsumption algorithm uses completion rules to modify a set of *labeled assertions*, until no new information can be added. An assertion is of the form (A, B) or (A, r, B) where $A, B \in \mathsf{N_C} \cup \{\top\}$ and $r \in \mathsf{R_C}$. We will consider a set of assertions $\mathcal{A}$ such that every assertion $a \in \mathcal{A}$ is labeled with a monotone Boolean formula $\mathsf{lab}(a)$. For brevity, if x is an axiom or an assertion, we will use the expression X^{ϕ} to denote that $\phi = \mathsf{lab}(x)$. The algorithm starts with the set of assertions $\mathcal{A}$ that contains $(A, \top)^{\mathsf{t}}, (A, A)^{\mathsf{t}}$ for every concept name A,[1] and uses the rules in Table 2 to extend $\mathcal{A}$. The rules consider as precondition a labeled axiom and a set of labeled assertions, and *insert* a labeled assertion a^{ψ} to $\mathcal{A}$: if $a \notin \mathcal{A}$, then we simply add a with label ψ to $\mathcal{A}$; otherwise, the assertion a is already in $\mathcal{A}$ with some label φ, and so we modify this label to $\mathsf{lab}(a) = \varphi \vee \psi$. The formula ψ appearing in the rules is built by the conjunction of all the formulas appearing in the precondition of the rule. Note that the rules are only applied if they really modify $\mathcal{A}$; i. e. if the assertion added by the rule is not yet in $\mathcal{A}$, or if the label of this assertion is modified to a strictly more general one.

The original subsumption algorithm for $\mathcal{EL}$ terminates in polynomial time [6]. It is also easy to see that the pinpointing extension of this algorithm is terminating, although the polynomial upper bound in execution time cannot be guaranteed. The algorithm adds always polynomially many assertions to the set $\mathcal{A}$, but each of these assertions may have its label repeatedly modified. Each time a label is modified, it is replaced by a more general formula, i. e. one that has more models. As there are exponentially many models, potentially the formula can be modified exponentially many times. Furthermore, in order to ensure termination, we need to test the equivalence of two formulas, which is a known NP-complete problem, and hence unlikely to be solvable in polynomial time. The following theorem, first presented in [5], summarizes the important properties of the pinpointing extension of the subsumption algorithm.

Theorem 1. *Given an $\mathcal{EL}$ TBox $\mathcal{T}$ in normal form, the pinpointing algorithm terminates in exponential time in the size of $\mathcal{T}$. After termination, the resulting set $\mathcal{A}$ of labeled assertions satisfies the following two properties for every*

[1] We use t to denote a propositional tautology, in order to avoid confusion with the $\mathcal{EL}$ constructor $\top$; analogously, we use f to denote the unsatisfiable formula.

$A, B \in \mathsf{N_C}$ *appearing in* $\mathcal{T}$*: (i)* $A \sqsubseteq_\mathcal{T} B$ *iff* $(A, B) \in \mathcal{A}$*, and (ii)* $\mathsf{lab}((A, B))$ *is a pinpointing formula for* $\mathcal{T}$ *w.r.t.* $A \sqsubseteq B$*.*

4 Automata-based Pinpointing

In [2] it was shown that, given an automata-based procedure that decides a property, it is possible to construct a *weighted* automaton, with its weights belonging to the lattice of monotone Boolean formulae, whose behaviour is the pinpointing formula. Furthermore, it was also shown that this behaviour can be computed using time that is polynomial in the number of states of the original automaton.

It turns out that the subsumption algorithm for $\mathcal{EL}$ can also be seen as an automata-based decision method. Indeed, the rule applications of this algorithm correspond to the bottom-up emptiness test of an appropriate automaton. Instead of describing first this automaton and then showing how it can be modified into a weighted automaton from which the pinpointing formula can be computed, we directly present the weighted variant. This weighted automaton will use the Büchi acceptance condition.

Let $\mathcal{T}$ be a TBox in normal form and $A_0, B_0 \in \mathsf{N_C}$. Once again we assume that every axiom $t \in \mathcal{T}$ is labeled with a unique propositional variable $\mathsf{lab}(t)$; abusing the notation, for every GCI in normal form that does not belong to $\mathcal{T}$, $t \notin \mathcal{T}$, we say that $\mathsf{lab}(t) = \mathsf{t}$ if t is $A \sqcap A \sqsubseteq A$ or $A \sqcap A \sqsubseteq \top$ for some $A \in \mathsf{N_C}$, and $\mathsf{lab}(t) = \mathsf{f}$ otherwise. We construct the weighted Büchi automaton $\mathcal{A}_{A_0,B_0,\mathcal{T}} = (Q, \mathsf{wt}, \mathsf{in}, F)$ over binary trees as follows:

- $Q = \{(A, B), (A, r, B) \mid A, B \in \mathsf{N_C}, r \in \mathsf{R_C}\}$,
- $\mathsf{in}((A_0, B_0)) = \mathsf{t}$, and $\mathsf{in}(q) = \mathsf{f}$ for all $q \neq (A_0, B_0)$,
- for all $A, B, B_1, B_2 \in \mathsf{N_C}$,
  - $\mathsf{wt}((A, B), (A, B_1), (A, B_2)) = \mathsf{lab}(B_1 \sqcap B_2 \sqsubseteq B)$,
  - $\mathsf{wt}((A, r, B), (A, B_1), (A, A)) = \mathsf{lab}(B_1 \sqsubseteq \exists r.B)$, and
  - $\mathsf{wt}((A, B), (A, r, B_1), (B_1, B_2)) = \mathsf{lab}(\exists r.B_2 \sqsubseteq B)$,
- $F = \{(A, A) \mid A \in \mathsf{N_C}\}$.

Given a state $q \in Q$, let $\mathsf{succ}(q)$ denote the set of all successful runs of $\mathcal{A}_{A_0,B_0,\mathcal{T}}$ whose root node is labeled with q. The behaviour of this automaton is then given by $\mathsf{in}((A_0, B_0)) \wedge \bigvee_{r \in \mathsf{succ}((A_0,B_0))} \mathsf{wt}(r)$, where the weight of a run r is the conjunction of the weights of the transitions of r. We have the following theorem.

Theorem 2. *Let* $\mathcal{T}$ *be a TBox in normal form and* $A_0, B_0 \in \mathsf{N_C}$*. The behaviour of* $\mathcal{A}_{A_0,B_0,\mathcal{T}}$ *is a pinpointing formula for* $\mathcal{T}$ *w.r.t.* $A_0 \sqsubseteq B_0$*.*

Notice that $\mathcal{A}_{A_0,B_0,\mathcal{T}}$ has polynomially many states in the size of $\mathcal{T}$, hence, the computation of its behaviour requires also polynomial time in the size of $\mathcal{T}$ [2].[2] Another important observation is that the same automaton can be used to compute the pinpointing formula for all possible subsumption relations between

[2] The results in [2] were presented only for the case of *looping* automata; it is nonetheless possible to adapt the same ideas to the more general case of Büchi automata [3].

concept names. Indeed, if we want to compute this formula for some $A' \sqsubseteq B'$, the automaton $\mathcal{A}_{A',B',\mathcal{T}}$ differs from $\mathcal{A}_{A_0,B_0,\mathcal{T}}$ only in the initial distribution in. The only purpose of this distribution is to limit the computation to the successful runs whose root is labeled with (A', B'), but does not affect the formula computed by the disjunction of the weights of those successful runs. The algorithm of [2] actually computes $\bigvee_{r \in \mathsf{succ}(q)} \mathsf{wt}(r)$ for every $q \in Q$, and thus also the pinpointing formula for every subsumption relation, just as the tableaux-based extension presented in the previous section does.

5 Conclusions

We have presented a tableau-based and an automata-based method for computing the pinpointing formula for an $\mathcal{EL}$ TBox w.r.t. a subsumption relation. Both approaches are extensions of the same algorithm for deciding subsumption [6]. We showed that the tableau-based algorithm has an exponential time worst case execution time, while the automata-based method terminates in polynomial time, measured in the size of the TBox. One possible reason for this discrepancy in execution time is the need of an NP-hard equivalence test in the tableau method, that is unnecessary for the automata approach. Additionally, the tableau-based extension allows the assertions to be added (and their labels be modified) in any arbitrary order, while the bottom-up algorithm for computing the behaviour of a weighted automaton yields a specific ordering through which the formula is constructed. One direction of future research is to use this insight to try to optimize other tableau-based pinpointing algorithms, through a rule-application ordering obtained by an automata-based method.

References

1. F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In L. P. Kaelbling and A. Saffiotti, editors, *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005)*, pages 364–369, Edinburgh (UK), 2005. Morgan Kaufmann, Los Altos.
2. F. Baader and R. Peñaloza. Automata-based axiom pinpointing. In A. Armando, P. Baumgartner, and G. Dowek, editors, *Proceedings of the International Joint Conference on Automated Reasoning (IJCAR 2008)*, volume 4667 of *LNAI*, pages 226–241, Sydney, Australia, 2008. Springer-Verlag.
3. F. Baader and R. Peñaloza. Automata-based axiom pinpointing. *Journal of Automated Reasoning*, 2009. Special Issue: IJCAR'08. Submitted.
4. F. Baader and R. Peñaloza. Axiom pinpointing in general tableaux. *Journal of Logic and Computation*, 2009. Special Issue: Tableaux'07. To appear.
5. F. Baader, R. Peñaloza, and B. Suntisrivaraporn. Pinpointing in the description logic $\mathcal{EL}^+$. In J. Hertzberg, M. Beetz, and R. Englert, editors, *Proceedings of the 30th German Annual Conference on Artificial Intelligence (KI'07)*, volume 4667 of *LNAI*, pages 52–67, Osnabrück, Germany, 2007. Springer-Verlag.
6. S. Brandt. Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and—what else? In R. L. de Mántaras and L. Saitta, editors, *Proc. of the 16th Eur. Conf. on Artificial Intelligence (ECAI 2004)*, pages 298–302, 2004.