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

Franz Baader[1], Rafael Peñaloza[2]*, and Boontawee Suntisrivaraporn[1]

[1] Theoretical Computer Science, TU Dresden, Germany
{baader,meng}@tcs.inf.tu-dresden.de
[2] Intelligent Systems, University of Leipzig, Germany
penaloza@informatik.uni-leipzig.de

1. Introduction

For a developer or user of a DL-based ontology, it is often quite hard to understand why a certain consequence holds, and even harder to decide how to change the ontology in case the consequence is unwanted. For example, in the current version of the medical ontology SNOMED [16], the concept *Amputation-of-Finger* is classified as a subconcept of *Amputation-of-Arm*. Finding the axioms that are responsible for this among the more than 350,000 terminological axioms of SNOMED without support by an automated reasoning tool is not easy.

As a first step towards providing such support, Schlobach and Cornet [14] describe an algorithm for computing all the *minimal subsets* of a given knowledge base that *have* a given *consequence*. In the following, we call such a set a *minimal axiom set (MinA)*. It helps the user to comprehend why a certain consequence holds. The knowledge bases considered in [14] are so-called unfoldable $\mathcal{ALC}$-terminologies, and the unwanted consequences are the unsatisfiability of concepts. The algorithm is an extension of the known tableau-based satisfiability algorithm for $\mathcal{ALC}$ [15], where labels keep track of which axioms are responsible for an assertion to be generated during the run of the algorithm. The authors also coin the name "axiom pinpointing" for the task of computing these minimal subsets.

The problem of computing MinAs of a DL knowledge base was actually considered earlier in the context of extending DLs by default rules. In [2], Baader and Hollunder solve this problem by introducing a labeled extension of the tableau-based consistency algorithm for $\mathcal{ALC}$-ABoxes [9], which is very similar to the one described later in [14]. The main difference is that the algorithm described in [2] does not directly compute minimal subsets that have a consequence, but rather a monotone Boolean formula whose variables correspond to the axioms of the knowledge bases and whose minimal satisfying valuations correspond to the MinAs.

The approach of Schlobach and Cornet [14] was extended by Parsia et al. [12] to more expressive DLs, and the one of Baader and Hollunder [2] was extended by Meyer et al. [11] to the case of $\mathcal{ALC}$-terminologies with general concept inclusions (GCIs), which are no longer unfoldable. Axiom pinpointing has also been considered in other research areas, though usually not under this name.

* Funded by the German Research Foundation (DFG) under grant GRK 446.

For example, in the SAT community, people have considered the problem of computing minimally unsatisfiable (and maximally satisfiable) subsets of a set of propositional formulae. The approaches for computing these sets developed there include special purpose algorithms that call a SAT solver as a black box [10, 5], but also algorithms that extend a resolution-based SAT solver directly [7, 18].

Whereas the previous work on pinpointing in DLs considered fairly expressive DLs that contain at least $\mathcal{ALC}$, this work is concerned with pinpointing in the inexpressive DL $\mathcal{EL}$, which has recently drawn considerable attention. On the one hand, several bio-medical ontologies such as SNOMED [16], the Gene Ontology [17], and large parts of Galen [13] can be expressed in $\mathcal{EL}$. On the other hand, reasoning in $\mathcal{EL}$ and some of its extensions remains polynomial even in the presence of GCIs [6, 1]. Although the polynomial-time subsumption algorithm for $\mathcal{EL}$ described in [6, 1] is not tableau-based, the ideas for extending tableau-based algorithms to pinpointing algorithms employed in [2, 14] can also be applied to this algorithm. However, we will see that the normalization phase employed by this algorithm introduces an additional problem. We will also consider the complexity of pinpointing in $\mathcal{EL}$. In contrast to the case of $\mathcal{ALC}$, where the subsumption problem is already highly complex, subsumption in $\mathcal{EL}$ is polynomial, which makes it easier to analyze the extent to which pinpointing is a source of additional complexity. Not surprisingly, it turns out that there may be exponentially many MinAs. In addition, even testing whether there is a MinA of cardinality $\leq n$ for a given natural number n is an NP-complete problem. However, one MinA can always be computed in polynomial time. Finally, we will provide some experimental results regarding the performance of a practical algorithm that computes one (not necessarily minimal) set that has a given consequence.

2. A pinpointing algorithm for $\mathcal{EL}$

Recall that $\mathcal{EL}$ allows for conjunction ($\sqcap$), existential restrictions ($\exists r.C$), and the top concept ($\top$). We consider general $\mathcal{EL}$-TBoxes $\mathcal{T}$ consisting of GCIs $C \sqsubseteq D$, where C, D are arbitrary $\mathcal{EL}$-concept descriptions, and are interested in the subsumption relation between concept names occurring in $\mathcal{T}$, which is denoted as $\sqsubseteq_\mathcal{T}$. Given such a TBox $\mathcal{T}$ and concept names A, B occurring in it, a *MinA for $\mathcal{T}$ w.r.t. $A \sqsubseteq B$* is a subset $\mathcal{S}$ of $\mathcal{T}$ such that $A \sqsubseteq_\mathcal{S} B$, but $A \not\sqsubseteq_{\mathcal{S}'} B$ for all strict subsets $\mathcal{S}' \subset \mathcal{S}$. For example, consider the TBox $\mathcal{T}$ consisting of the GCIs

$$\mathrm{ax}_1\colon\ A \sqsubseteq \exists r.A, \qquad \mathrm{ax}_2\colon\ A \sqsubseteq Y, \qquad \mathrm{ax}_3\colon\ \exists r.Y \sqsubseteq B, \qquad \mathrm{ax}_4\colon\ Y \sqsubseteq B. \qquad (1)$$

We have $A \sqsubseteq_\mathcal{T} B$, and it is easy to see that $\{\mathrm{ax}_2, \mathrm{ax}_4\}$ and $\{\mathrm{ax}_1, \mathrm{ax}_2, \mathrm{ax}_3\}$ are the MinAs of $\mathcal{T}$ w.r.t. $A \sqsubseteq B$.

In the following, we show how the polynomial-time subsumption algorithm for $\mathcal{EL}$ with GCIs [6, 1] can be modified to a pinpointing algorithm. However, instead of computing the MinAs directly, we follow the approach introduced in [2] that computes a monotone Boolean formula from which the MinAs can be derived. To define this formula, which we will call a pinpointing formula in the following, we assume that every GCI $t \in \mathcal{T}$ is labeled with a unique propositional variable, $lab(t)$. Let $lab(\mathcal{T})$ be the set of all propositional variables labeling GCIs in $\mathcal{T}$. A *monotone Boolean formula* over $lab(\mathcal{T})$ is a Boolean formula using (some of) the variables in $lab(\mathcal{T})$ and only the binary connectives conjunction and disjunction and the nullary connective `t` (for truth). As usual, we identify a propositional *valuation* with the set of propositional variables it makes true. For a valuation $\mathcal{V} \subseteq lab(\mathcal{T})$, let $\mathcal{T}_\mathcal{V} := \{t \in \mathcal{T} \mid lab(t) \in \mathcal{V}\}$.

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

In our example, we can take $lab(\mathcal{T}) = \{\mathrm{ax}_1, \ldots, \mathrm{ax}_4\}$ as set of propositional variables. It is easy to see that $\mathrm{ax}_2 \wedge (\mathrm{ax}_4 \vee (\mathrm{ax}_1 \wedge \mathrm{ax}_3))$ is a pinpointing formula for $\mathcal{T}$ w.r.t. $A \sqsubseteq B$.

Let ϕ be a pinpointing formula for $\mathcal{T}$ w.r.t. $A \sqsubseteq B$. If we order valuations by set inclusion, then we obviously have that

$$\{\mathcal{T}_\mathcal{V} \mid \mathcal{V} \text{ is a minimal valuation satisfying } \phi\}$$

is the set of all MinAs for $\mathcal{T}$ w.r.t. $A \sqsubseteq B$. This shows that it is enough to design an algorithm for computing a pinpointing formula to obtain all MinAs. For example, one possibility is to bring ϕ into disjunctive normal form and then remove disjuncts implying other disjuncts. Note that this may cause an exponential blowup, which means that, in some cases, the pinpointing formula provides us with a compact representation of the set of all MinAs. Also note that this blowup is not really in the size of the pinpointing formula but rather in the number of variables. Thus, if the size of the pinpointing formula is already exponential in the size of the TBox $\mathcal{T}$ (which may well happen), computing all MinAs from it is still "only" exponential in the size of $\mathcal{T}$.

In order to describe our algorithm for computing pinpointing formulae for subsumption in $\mathcal{EL}$, we must briefly recall the subsumption algorithm for $\mathcal{EL}$ [6, 1]. First, this algorithm transforms a given TBox into a *normal form* where all GCIs have one of the following forms: $A_1 \sqcap \ldots \sqcap A_n \sqsubseteq B$, $A \sqsubseteq \exists r.B$, $\exists r.A \sqsubseteq B$, where $n \geq 1$ and $A, A_1, \ldots, A_n, B$ are concept names (including $\top$). This transformation can be achieved in linear time using simple transformation rules, which basically break down complex GCIs into simpler ones (see [6, 1] for details). For the pinpointing extension it is relevant that the relationship between original axioms and normalized axioms is many to many: one axiom in the original TBox can give rise to several axioms in the normalized one, and one axiom in the normalized TBox can come from several axioms in the original TBox. For example, consider the GCIs $A \sqsubseteq B_1 \sqcap B_2, A \sqsubseteq B_2 \sqcap B_3$, which are normalized to $A \sqsubseteq B_1, A \sqsubseteq B_2, A \sqsubseteq B_3$. Each original GCI gives rise to two normalized ones, and the normalized GCI $A \sqsubseteq B_2$ has two sources, i.e., it is present in the normalized TBox if the first *or* the second original GCI is present in the input TBox.

If $A_1 \sqcap \ldots \sqcap A_n \sqsubseteq B \in \mathcal{T}$	**and** $\{(X, A_1), \ldots, (X, A_n)\} \subseteq \mathcal{A}$	**then** add (X, B) to $\mathcal{A}$.
If $A \sqsubseteq \exists r.B \in \mathcal{T}$	**and** $(X, A) \in \mathcal{A}$	**then** add (X, r, B) to $\mathcal{A}$.
If $\exists r.A \sqsubseteq B \in \mathcal{T}$	**and** $\{(X, r, Y), (Y, A)\} \subseteq \mathcal{A}$	**then** add (X, B) to $\mathcal{A}$.

Fig. 1. Completion rules for subsumption in $\mathcal{EL}$.

Given a TBox $\mathcal{T}$ in normal form, *the subsumption algorithm for $\mathcal{EL}$* employs completion rules to extend an initial set of assertions until no more assertions can be added. Assertions are of the form (A, B) or (A, r, B) where A, B are concept names occurring in $\mathcal{T}$ or $\top$, and r is a role name occurring in $\mathcal{T}$. Intuitively, the assertion (A, B) expresses that $A \sqsubseteq_{\mathcal{T}} B$ holds and (A, r, B) expresses that $A \sqsubseteq_{\mathcal{T}} \exists r.B$ holds. The algorithm starts with a set of assertions $\mathcal{A}$ that contains $(A, \top)$ and (A, A) for every concept name A, and then uses the rules shown in Fig. 1 to extend $\mathcal{A}$. Note that such a rule is only applied if it really extends $\mathcal{A}$, i.e., if the assertion added by the rule is not yet contained in $\mathcal{A}$. As shown in [6, 1], rule application terminates after a polynomial number of steps, and the set of assertions $\mathcal{A}$ obtained after termination satisfies $A \sqsubseteq_{\mathcal{T}} B$ iff $(A, B) \in \mathcal{A}$ for all concept names A, B occurring in $\mathcal{T}$.

In *the pinpointing extension* of this algorithm, assertions a are also labeled with monotone Boolean formulae $\mathit{lab}(a)$. The initial assertions $(A, \top)$ and (A, A) receive label `t`. The definition of rule application is modified as follows. Assume that the precondition of a rule from Fig. 1 are satisfied for the set of assertions $\mathcal{A}$ w.r.t. the TBox $\mathcal{T}$. Let ϕ be the conjunction of the labels of the GCIs from $\mathcal{T}$ and the assertions from $\mathcal{A}$ occurring in the precondition. If the assertion in the consequence of the rule does not yet belong to $\mathcal{A}$, then it is added with label ϕ. If the assertion is already there with label ψ, then its label is changed to $\psi \vee \phi$ if this formula is not equivalent to ψ; otherwise (i.e., if ϕ implies ψ) the rule is not applied.

It is easy to see that this modified algorithm always terminates, though not necessarily in polynomial time. In fact, there are polynomially many assertions that can be added to $\mathcal{A}$. If the label of an assertion is changed, then the new label is a more general monotone Boolean formula, i.e., it has more models than the original label. Since there are only exponentially many models, the label of a given assertion can be changed only exponentially often. In addition, the set of assertions $\mathcal{A}$ obtained after termination is identical to the one obtained by the unmodified algorithm, and for all assertions $(A, B) \in \mathcal{A}$ we have that $\mathit{lab}((A, B))$ is a pinpointing formula for $\mathcal{T}$ w.r.t. $A \sqsubseteq B$.

As an example, consider the TBox consisting of the GCIs given in (1). The pinpointing algorithm proceeds as follows. Since $\mathrm{ax}_2 : A \sqsubseteq Y \in \mathcal{T}$ and $(A, A) \in \mathcal{A}$ with label `t`, the assertion (A, Y) is added to $\mathcal{A}$ with label ax_2 (actually with label $\mathrm{ax}_2 \wedge$ `t`, which is equivalent to ax_2). Since $\mathrm{ax}_1 : A \sqsubseteq \exists r.A \in \mathcal{T}$ and $(A, A) \in \mathcal{A}$ with label `t`, (A, r, A) is added to $\mathcal{A}$ with label ax_1. Since $\mathrm{ax}_4 : Y \sqsubseteq B \in \mathcal{T}$ and $(A, Y) \in \mathcal{A}$ with label ax_2, (A, B) is added to $\mathcal{A}$ with label $\mathrm{ax}_2 \wedge \mathrm{ax}_4$. Finally, since $\mathrm{ax}_3 : \exists r.Y \sqsubseteq B \in \mathcal{T}$, $(A, Y) \in \mathcal{A}$ with label ax_2, and $(A, r, A) \in \mathcal{A}$ with label ax_1, the label of $(A, B) \in \mathcal{A}$ is modified from $\mathrm{ax}_2 \wedge \mathrm{ax}_4$

Algorithm 1 Compute one MinA for $\mathcal{T} = \{t_1, \ldots, t_n\}$ w.r.t. $A \sqsubseteq B$.

```
1: if $A \not\sqsubseteq_{\mathcal{T}} B$ then
2:     return no MinA
3: $\mathcal{S} := \mathcal{T}$.
4: for $1 \leq i \leq n$ do
5:     if $A \sqsubseteq_{\mathcal{S} \setminus \{t_i\}} B$ then
6:         $\mathcal{S} := \mathcal{S} \setminus \{t_i\}$
7: return $\mathcal{S}$
```

to $(\mathrm{ax}_2 \wedge \mathrm{ax}_4) \vee (\mathrm{ax}_1 \wedge \mathrm{ax}_2 \wedge \mathrm{ax}_3)$. This final label of (A, B) is a pinpointing formula for $\mathcal{T}$ w.r.t. $A \sqsubseteq B$.

As described until now, our pinpointing algorithm for $\mathcal{EL}$ can only deal with normalized TBoxes, i.e., the pinpointing formula ϕ it yields contains propositional variables corresponding to the normalized GCIs. However, modifying ϕ to a *pinpointing formula for the original TBox* is quite simple. Assume that the original GCIs are also associated with propositional variables. Each normalized GCI has a finite number of original GCIs as sources. We modify ϕ by replacing the propositional variable for each normalized axiom by the disjunction of the propositional variables of its sources.

3. The complexity of pinpointing in $\mathcal{EL}$

If we want to compute all MinAs, then in the worst case an exponential runtime cannot be avoided since there may be *exponentially many MinAs* for a given TBox. This is shown by the following example.

Example 1. For all $n \geq 1$, the size of the TBox

$$\mathcal{T}_n := \{B_{i-1} \sqsubseteq P_i \sqcap Q_i, P_i \sqsubseteq B_i, Q_i \sqsubseteq B_i \mid 1 \leq i \leq n\}$$

is linear in n, and we have $B_0 \sqsubseteq_{\mathcal{T}_n} B_n$. There are 2^n MinAs for $\mathcal{T}_n$ w.r.t. $B_0 \sqsubseteq B_n$ since, for each $i, 1 \leq i \leq n$, it is enough to have $P_i \sqsubseteq B_i$ or $Q_i \sqsubseteq B_i$ in the set.

On the other hand, a *single MinA can be computed in polynomial time* by the simple Algorithm 1, which goes through all GCIs (in a given fixed order) and throws away those that are not needed to obtain the desired subsumption relationship. Since the algorithm performs $n + 1$ subsumption tests (where n is the cardinality of $\mathcal{T}$), and each such test takes only polynomial time, the overall complexity of this algorithm is polynomial. It is easy to see that its output (in case $A \sqsubseteq_{\mathcal{T}} B$) is indeed a MinA for $\mathcal{T}$ w.r.t. $A \sqsubseteq B$.

However, as soon as we want to know more about the properties of the set of *all* MinAs, this cannot be achieved in polynomial time (unless P=NP). For example, the following *minimum cardinality problem* is NP-complete: given an $\mathcal{EL}$-TBox $\mathcal{T}$, concept names A, B occurring in $\mathcal{T}$, and a natural number n, is there a MinA for $\mathcal{T}$ w.r.t. $A \sqsubseteq B$ of cardinality $\leq n$? The problem is in NP, since one can simply guess a subset $\mathcal{S}$ of $\mathcal{T}$ cardinality n, and then check in polynomial time whether $A \sqsubseteq_{\mathcal{S}} B$. Clearly, such a set exists iff there is a MinA of cardinality $\leq n$.

NP-hardness can be shown by a reduction of the NP-hard *hitting set problem* [8]: given a collection $S_1, \ldots S_k$ of sets and a natural number n, is there a set S of cardinality $\leq n$ such that $S \cap S_i \neq \emptyset$ for $i = 1, \ldots, k$. Such a set S is called a *hitting set*. In the reduction, we use a concept name P for every element $p \in S_1 \cup \ldots \cup S_n$ as well as the additional concept names $A, B, Q_1, \ldots, Q_k$. Given $S_1 = \{p_{11}, \ldots, p_{1\ell_1}\}, \ldots, S_k = \{p_{k1}, \ldots, p_{k\ell_k}\}$, we define the TBox

$$\begin{aligned}\mathcal{T} := \{&P_{ij} \sqsubseteq Q_i \mid 1 \leq i \leq k, 1 \leq j \leq \ell_i\} \cup \\ &\{A \sqsubseteq P_{ij} \mid 1 \leq i \leq k, 1 \leq j \leq \ell_i\} \cup \{Q_1 \sqcap \ldots \sqcap Q_k \sqsubseteq B\}.\end{aligned}$$

It is easy to see that $S_1, \ldots, S_k$ has a hitting set of cardinality $\leq n$ iff there is a MinA for $\mathcal{T}$ w.r.t. $A \sqsubseteq B$ of cardinality $\leq n + k + 1$.

4. A practical algorithm for computing one MinA

Although it requires only polynomial time, computing one MinA using Algorithm 1 may still be impractical for very large TBoxes like SNOMED. In fact, the algorithm has to make as many calls of the subsumption algorithm as there are axioms in the TBox (in the case of SNOMED, more than 350,000). Here we propose an *improved algorithm* that proceeds in two steps: (i) first compute a (not necessarily minimal) subset of the TBox from which the subsumption relationship follows; (ii) then minimize this set using Algorithm 1. Of course, this approach makes sense only if the algorithm used in step (i) is efficient and produces fairly small sets. It would not help to use the trivial algorithm that always produces the whole TBox.

An algorithm that realizes step (i) and runs in polynomial time can easily be obtained from the pinpointing algorithm sketched in Section 2. The only modification is the following: if an assertion in the consequence of a rule already belongs to the current set of assertions, then this rule is not applied, i.e., once an assertion is there with some label, the label remains unchanged. Thus, every assertion (A, B) in the final set has a conjunction of propositional variables as its label, which clearly corresponds to a subset of the TBox from which the subsumption relationship $A \sqsubseteq B$ follows. In general, this subset is not minimal, however. (Because of the space constraints, we cannot give an example demonstrating this.)

As described until now, this modified algorithm works on normalized TBoxes. To get an appropriate subset of the original axioms, one can use a greedy strategy for producing a set of original axioms that covers a given set $\mathcal{S}$ of normalized axioms in the following sense. For each original axiom t, let $\mathcal{S}_t$ be the set of normalized axioms t gives rise to. The set $\mathcal{T}'$ of original axioms *covers* $\mathcal{S}$ if $\mathcal{S} \subseteq \bigcup_{t \in \mathcal{T}'} \mathcal{S}_t$. The use of a greedy strategy adds another possible source of non-minimality. (We use a non-optimal greedy strategy to keep the algorithm polynomial. In fact, even determining whether there is a cover set of size $\leq n$ is another NP-complete problem [8].)

Our preliminary experimental results confirm that this algorithm is indeed more practical than Algorithm 1. Based on the CEL reasoner [3], we have implemented a slightly extended version of the practical algorithm for computing one MinA in $\mathcal{EL}^+$, which is $\mathcal{EL}$ extended with complex role inclusions, and thus can express role hierarchies and transitive roles. The experiments were run on a variant of the Galen Medical Knowledge Base [13],[3] which is a TBox consisting of more than 4,000 axioms. On the normalized version of this TBox, CEL needs about 14 sec to compute all subsumption relationships between concept names occurring in this TBox. Overall, more than 27,000 subsumption relationships are computed. The overhead for computing for all of these subsumption relationships (possibly non-minimal) subsets from which they already follow was a bit more than 50%: the modified pinpointing algorithm described above needed about 23 sec. Going from the subsets of the normalized TBox to the corresponding (still possibly non-minimal) subsets of the original Galen TBox took 0.27 sec. Finally, the overall time required for minimizing these sets using Algorithm 1 (with CEL as the subsumption reasoner) was 9:45 min. For these last two numbers one should take into account, however, that these involved treating more then 27,000 such sets. For a single such set, the average post-processing time was negligible (on average 21 milliseconds). Also note that applying Algorithm 1 directly to the whole TBox for just one subsumption relationship (between *Renal-Artery* and *Artery-Which-Has-Laterality*) took more than 7 hours.

Thus, from the point of view of runtime, our practical algorithm behaves quite well on Galen. The same can be said about the quality of its results. The average size of an axiom set computed by the algorithm before using Algorithm 1 to minimize it was 5 (with maximum size 31), which is quite small and thus means that this set can directly be given to the user as an explanation for the subsumption relationship. Also, the computed sets were almost minimal: on average, the possibly non-minimal sets computed by the algorithm were only 2.59% larger than the minimal ones. When considering the normalized TBox (i.e., without translating back to the original TBox), this number was even better (0.1%). This means that in most cases it is probably not necessary to further minimize the sets using Algorithm 1. If demanded by the user for a specific subsumption relationship it can still be done without taking much time.

5. Additional work on pinpointing

The pinpointing extension of the subsumption algorithm for $\mathcal{EL}$ described in Section 2 as well as the pinpointing algorithm for $\mathcal{ALC}$ described in [2] are instances of a general approach for modifying "tableau-like" reasoning procedures to pinpointing procedures [4].

Instead of computing minimal subsets that have a given consequence, one sometimes also wants to compute maximal subsets that do not have a given consequence. Given the pinpointing formula ϕ, these sets correspond to maximal

[3] Since Galen uses expressivity not available in $\mathcal{EL}^+$, we have simplified it by removing inverse role axioms and treating functional roles as ordinary ones.

valuations that do not satisfy ϕ. The complexity results from Section 3 hold accordingly for such maximal sets. However, we currently do not know how to obtain a practical algorithm computing one such set (i.e., the results of Section 4 cannot be transferred to the case of maximal sets).

References

[1] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. of IJCAI 2005*, pages 364–369. Morgan Kaufmann, Los Altos, 2005.

[2] F. Baader and B. Hollunder. Embedding defaults into terminological knowledge representation formalisms. *J. of Automated Reasoning*, 14:149–180, 1995.

[3] F. Baader, C. Lutz, and B. Suntisrivaraporn. CEL—a polynomial-time reasoner for life science ontologies. In *Proceedings of IJCAR'06*, volume 4130 of *LNAI*, pages 287–291. Springer-Verlag, 2006.

[4] F. Baader and R. Penaloza. Axiom pinpointing in general tableaux. LTCS-Report 07-01, Germany, 2007. See http://lat.inf.tu-dresden.de/research/reports.html.

[5] J. Bailey and P.J. Stuckey. Discovery of minimal unsatisfiable subsets of constraints using hitting set dualization. In *Proc. of PADL'05*, volume 3350 of *LNCS*, pages 174–186. Springer, 2005.

[6] S. Brandt. Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and—what else? In *Proc. of ECAI 2004*, pages 298–302.

[7] G. Davydov, I. Davydova, and H. Kleine Büning. An efficient algorithm for the minimal unsatisfiability problem for a subclass of CNF. *Ann. of Mathematics and Artificial Intelligence*, 23(3–4):229–245, 1998.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability — A guide to NP-completeness.* W. H. Freeman and Company, San Francisco (CA, USA), 1979.

[9] B. Hollunder. Hybrid inferences in KL-ONE-based knowledge representation systems. In *Proc. of German Workshop on AI*, pages 38–47. Springer, 1990.

[10] M.H. Liffiton and K.A. Sakallah. On finding all minimally unsatisfiable subformulas. In *Proc. of SAT'05*, volume 3569 of *LNCS*, pages 173–186. Springer, 2005.

[11] T. Meyer, K. Lee, R. Booth, and J. Z. Pan. Finding maximally satisfiable terminologies for the description logic $\mathcal{ALC}$. In *Proc. of AAAI 2006*. AAAI Press.

[12] B. Parsia, E. Sirin, and A. Kalyanpur. Debugging OWL ontologies. In *Proc. of WWW'05*, pages 633–640. ACM, 2005.

[13] A. Rector and I. Horrocks. Experience building a large, re-usable medical ontology using a description logic with transitivity and concept inclusions. In *Proceedings of AAAI'97*, Stanford, CA, 1997. AAAI Press.

[14] S. Schlobach and R. Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *Proc. of IJCAI 2003*, pages 355–362, Acapulco, Mexico, 2003. Morgan Kaufmann, Los Altos.

[15] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[16] K.A. Spackman, K.E. Campbell, and R.A. Cote. SNOMED RT: A reference terminology for health care. *J. of the Am. Med. Inf. Ass.*, pages 640–644, 1997.

[17] The Gene Ontology Consortium. Gene Ontology: Tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[18] L. Zhang and S. Malik. Validating SAT solvers using an independent resolution-based checker: Practical implementations and other applications. In *Proc. of DATE'03*, pages 10880–10885. IEEE Computer Society Press, 2003.