Algorithmic Aspects of Herbrand Models Represented by Ground Atoms with Ground Equations

Bernhard Gramlich and Reinhard Pichler*

Technische Universität Wien

Abstract. *Automated model building* has evolved as an important subdiscipline of *automated deduction* over the past decade. One crucial issue in automated model building is the selection of an appropriate (finite) representation of (in general infinite) models. Quite a few such formalisms have been proposed in the literature. In this paper, we concentrate on the representation of Herbrand models by ground atoms with ground equations (GAE-models), introduced in [9]. For the actual work with any model representation, efficient algorithms for two decision problems are required, namely: The *clause evaluation problem* (i.e.: Given a clause C and a representation $\mathcal{M}$ of a model, does C evaluate to "true" in this model?) and the *model equivalence problem* (i.e.: Given two representations $\mathcal{M}_1$ and $\mathcal{M}_2$, do they represent the same model?). Previously published algorithms for these two problems in case of GAE-models require exponential time. We prove that the clause evaluation problem is indeed intractable (that is, coNP-complete), whereas the model equivalence problem can be solved in polynomial time. Moreover, we show how our new algorithm for the model equivalence problem can be used to transform an arbitrary GAE-model into an equivalent one with better computational properties.

1 Introduction

One of the principal goals of automated deduction is the construction of algorithms to detect the validity of a first-order formula (or, equivalently, the unsatisfiability of a corresponding clause set). The use of *models* in this activity can be essentially twofold: First, by constructing a counter-model (rather than just answering "NO"), the user not only learns *that* a given formula is not a theorem but also gets some insight as to *why* it is not a theorem. Second, models can be used to speed up the deduction process itself, e.g. by taking a model of parts of a given clause set as an input to a theorem prover based on semantic resolution or semantic clash resolution. The former use of models was the main motivation for early attempts towards automated model building (cf. [22]). The latter one was studied e.g. in [3]. Many research activities in this area over the past years

* TU Wien, Inst. f. Computersprachen, AG Theoretische Informatik und Logik, Favoritenstrasse 9 – E185/2, A-1040 Wien, Austria, {gramlich|reini}@logic.at

demonstrate the increasing interest in this subfield of automated deduction (cf. e.g. [4], [21], [19,13], [5], [23], [3], [8,9], [14,15], [2]).

A crucial issue in automated model building is the selection of an appropriate representation of models. Quite a few such formalisms have been proposed in the literature, e.g.: linear atomic representations, general atomic representations, literals with equational constraints, finite tree automata, regular term grammars, term schematizations via R-terms or I-terms, ground atoms with ground equations. A comprehensive overview of these formalisms and a comparison of their expressive power is given in [14,15] (cf. also [6]). In this work, we concentrate on the representation of models[1] by ground atoms with ground equations (GAE-models, for short) introduced in [9]. A GAE-model $\mathcal{M}$ over some finite signature Σ, consisting of a set $\mathcal{F}$ of constant and function symbols and a (disjoint) set $\mathcal{P}$ of uninterpreted predicate symbols ($\Sigma = \mathcal{F} \uplus \mathcal{P}$), is given through a finite set $\mathcal{A} = \{A_1, \ldots, A_m\}$ of ground atoms and a finite set $\mathcal{E} = \{s_1 = t_1, \ldots, s_n = t_n\}$ of ground equations over $\mathcal{F}$. A ground atom B over Σ evaluates to "true" in the model $\mathcal{M}$ iff B is equal to some atom $A_i \in \mathcal{A}$ in the equational theory defined by $\mathcal{E}$.[2]

Example 1. Let $\mathcal{C} = \{P(x) \vee P(f(x)), \neg P(x) \vee \neg P(f(x))\}$ be a clause set over the signature $\Sigma = \{P, f, a\}$. Then $\mathcal{M} = (\mathcal{A}, \mathcal{E})$ with $\mathcal{A} = \{P(a)\}$ and $\mathcal{E} = \{a = f(f(a))\}$ is a model of the clause set $\mathcal{C}$. Note that the set of ground atoms that are "true" in $\mathcal{M}$ is $\{P(a), P(f^2(a)), P(f^4(a)), \ldots\}$. Another model of $\mathcal{C}$ would be $\mathcal{M}' = (\mathcal{A}', \mathcal{E}')$ with $\mathcal{A}' = \{P(f(a))\}$ and $\mathcal{E}' = \{a = f(f(a))\} = \mathcal{E}$. In this case, the set of ground atoms that evaluate to "true" in $\mathcal{M}'$ is $\{P(f(a)), P(f^3(a)), P(f^5(a)), \ldots\}$.

For the actual work with any model representation formalism, efficient algorithms for the following decision problems are crucial:

- *Clause Evaluation*: Given a clause C and a model representation $\mathcal{M}$, does C evaluate to "true" in the model represented by $\mathcal{M}$.
- *Model Equivalence*: Given two model representations $\mathcal{M}_1$ and $\mathcal{M}_2$, do they represent the same model, i.e.: Does every clause (over the given signature) have the same truth value in both models.

The importance of the first decision problem is obvious. Without an efficient clause evaluation procedure, semantic resolution or semantic clash resolution based on such models would not be feasible. The importance of the latter decision problem comes from the following observation: In general, the cost of clause

[1] For simplicity, but slightly abusing common terminology, throughout the paper we will speak of *models* rather than of *interpretations* sometimes tacitly assuming there is some (implicitly given) clause set for which the considered interpretation is indeed a model.

[2] Note that in contrast to [9] our notion of GAE-model uses (ground) equality only as a representation formalism but doesn't allow (ground) equations as (ground) atoms. The results, however, can be partially extended to cover this case, too (cf. Section 6).

evaluation heavily depends on the *representation* of the model rather than just on the model itself. We shall see that also in case of GAE-models, it is important to look for transformations of a given model representation into an equivalent one with "better" computational properties. But then we have to make sure, of course, that the model representation resulting from such a transformation is equivalent to the original one.

In [9], a clause evaluation procedure for GAE-models was presented. The computational complexity of this procedure was not explicitly analyzed, but it is clearly exponential. In [14], it is shown that GAE-models have the same expressive power as finite tree automata and regular tree grammars. In particular, a transformation of GAE-models into equivalent finite tree automata and vice versa is given there. Hence, one way to decide the equivalence of two GAE-models $\mathcal{M}_1$ and $\mathcal{M}_2$ is to construct equivalent finite tree automata A_1 and A_2, respectively, and to check whether they generate the same languages L_1 and L_2. Yet, the naive approach of proceeding like that yields an exponential time algorithm ([14, 15, 18]), cf. the detailed discussion at the end of Section 4.

In this paper, we give a new algorithm for the model equivalence problem, which works in polynomial time. As to clause evaluation, we cannot expect to do equally well by the coNP-completeness, that we are going to prove here. However, we shall provide a transformation of an arbitrary GAE-model into an equivalent one with better computational properties. In many cases, this should lead to a significant speed-up of clause evaluation, even though the exponential worst-case complexity can clearly not be avoided.

The remainder of this work is organized as follows: In Section 2 we recall some basic notions and results on ground term rewrite systems and on GAE-models. In Sections 3 and 4, we analyze the complexity of the clause evaluation problem and we provide a new algorithm for the model equivalence problem of GAE-models as well as a detailed complexity analysis and comparison with tree automata based approaches. In Section 5 we suggest a transformation of GAE-models into computationally better ones via the model equivalence algorithm of Section 4. Finally, in Section 6, we briefly discuss the obtained results and related topics. Due to lack of space, some proofs have been omitted.[3]

2 Preliminaries

2.1 Ground Term Rewrite Systems

We assume familiarity with the basic terminology and theory in (ground) term rewriting, cf. e.g. [1], [7]. For the sake of readability we just recall a few notions needed. A term rewrite system (TRS) $\mathcal{R}$ is *convergent* iff it is *confluent* and *terminating*. $\mathcal{R}$ is *canonical* iff it is convergent and *(inter)reduced*[4] . The *normal form* of a term t w.r.t. some (convergent) TRS $\mathcal{R}$ is denoted by $NF_{\mathcal{R}}(t)$. The *size*

[3] Complete proofs are given in the full version [12] of this paper.

[4] This means that all right-hand sides of rules in $\mathcal{R}$ are irreducible and the left-hand side of every rule $l \to r$ in $\mathcal{R}$ is irreducible w.r.t. $\mathcal{R} \setminus \{l \to r\}$.

$|t|$ and *depth* $\tau(t)$ of t are defined as usual. A ground term rewrite system (GTRS) is a TRS without variables in its rules. It is well-known, that any finite system of *ground* term equations (GTES) can be turned into a (logically) equivalent canonical finite GTRS. This can even be achieved quite efficiently, namely in polynomial time (w.r.t. the size of the original system). The earlier *completion* algorithms for that purpose are in essence *congruence closure* based methods (cf. e.g. [10]) where the original signature is extended by fresh constants in order to name certain congruence classes. Yet, direct polynomial completion without extending signatures is also possible via a more sophisticated approach as demonstrated in [17] (cf. also [20]).

Next we summarize those basic results that will frequently be used in subsequent sections (cf. [17], [20]):

(a) Any finite GTES $\mathcal{E}$ can be completed into a canonical GTRS $\mathcal{R}$ (over the same signature) in polynomial (more precisely: quadratic) time (w.r.t. the size of $\mathcal{E}$). Moreover, the size of the completed system is bounded by the size of the original one: $|\mathcal{R}| \leq |\mathcal{E}|$.
(b) If, by a canonical GTRS $\mathcal{R}$, a term t reduces at the root to t', then t' is in normal form.[5]
(c) Given a term t and a canonical GTRS $\mathcal{R}$, computing $NF_{\mathcal{R}}(t)$ can be done in at most $|t|$ steps.[6]
(d) Let $\mathcal{M} = (\mathcal{A}, \mathcal{E})$ be a GAE-model over some Herbrand universe $H = \mathcal{T}(\mathcal{F})$ with signature Σ ($\mathcal{F} \subseteq \Sigma$), s.t. $\mathcal{A}$ is a finite set of ground atoms over Σ, $\mathcal{E}$ is a finite set of ground equations over $\mathcal{F}$, and $\mathcal{U}$ is the set of all terms and subterms occurring in $\mathcal{A} \cup \mathcal{E}$. Then it is possible to compute the triple $(\mathcal{R}, \widehat{\mathcal{A}}, \widehat{\mathcal{U}})$ in quadratic time, where $\mathcal{R}$ is a canonical GTRS equivalent to $\mathcal{E}$, $\widehat{\mathcal{A}}$ is the set of normal forms w.r.t. $\mathcal{R}$ of the atoms in $\mathcal{A}$ and $\widehat{\mathcal{U}}$ is the set of normal forms w.r.t. $\mathcal{R}$ of $\mathcal{U}$.[7]
(e) If $t \in H$ normalizes into $\widehat{\mathcal{U}}$, then every subterm of t also has its normal form in $\widehat{\mathcal{U}}$. Hence, in particular, $\widehat{\mathcal{U}}$ is closed w.r.t. subterms, i.e.: If $t \in \widehat{\mathcal{U}}$, then every subterm of t is in $\widehat{\mathcal{U}}$.[8]

2.2 GAE-Models

The representation of models by ground atoms with ground equations (GAE-models, for short) was introduced in [9] in order to provide a model construction algorithm for a certain syntactical class of clauses, the so-called class $\mathrm{PVD}_g^=$. Actually, GAE-models are a powerful mechanism which allow us to represent

[5] Obvious, since $\mathcal{R}$ is interreduced.
[6] Note that contracting a subterm in t makes the corresponding subterm in the resulting term irreducible. Hence, the statement follows by (b).
[7] This is a consequence of (a)-(c), where, in order to guarantee the complexity bound we need to employ a ground total reduction ordering for completion that is size decreasing.
[8] Property (e) follows from the subterm closedness of $\mathcal{U}$ and the fact that $\mathcal{R}$ is canonical. The details are worked out in the full version [12] of the paper.

models of clause sets where other formalisms would fail, e.g.: The clause set $\mathcal{C} = \{P(x) \vee P(f(x)), \neg P(x) \vee \neg P(f(x))\}$ in Example 1 has a simple GAE-model, whereas it is impossible to represent a model of $\mathcal{C}$ via atomic representations or even via the much more powerful formalism of literals with equational constraints (cf. [16]).

In this section, we briefly recall the clause evaluation algorithm from [9] for GAE-models. It has already been mentioned in Section 2.1, that the evaluation of ground atoms is done via a canonical GTRS, i.e.: Let a GAE-model $\mathcal{M}$ be given through a set of ground atoms $\mathcal{A}$ and a set of ground equations $\mathcal{E}$. Moreover, let $\mathcal{R}$ be a canonical GTRS equivalent to $\mathcal{E}$ and let $\widehat{\mathcal{A}}$ denote the set of normal forms of $\mathcal{A}$ w.r.t. $\mathcal{R}$. Then a ground atom B evaluates to "true" in $\mathcal{M}$, iff $NF_{\mathcal{R}}(B) \in \widehat{\mathcal{A}}$. The evaluation of ground clauses $B_1 \vee \ldots \vee B_k \vee \neg B_{k+1} \vee \ldots \vee \neg B_l$ is then also obvious.

Now let $\mathcal{U}$ denote the set of all terms and subterms occurring in $\mathcal{A}$ and $\mathcal{E}$. Moreover, let $\widehat{\mathcal{U}}$ denote the set of normal forms w.r.t. $\mathcal{R}$ of the terms in $\mathcal{U}$. The key to the evaluation of a non-ground clause C are the following lessons learned from the evaluation of a ground atom B:

1. Any subterm u occurring in B may be replaced by its normal form $NF_{\mathcal{R}}(u)$ without affecting the truth value of B. Hence, suppose that all terms of H can be normalized into $\widehat{\mathcal{U}}$. Then it is not necessary to check all possible ground instances $C\vartheta$ of C, that are obtained by substituting all possible terms in H for the variables in C. Instead, it suffices to check the truth value of those ground instances $C\vartheta$ of C, where the range of ϑ is restricted to the *finite* set $\widehat{\mathcal{U}}$.
2. Suppose that some subterm u occurring in B does not normalize into $\widehat{\mathcal{U}}$. In this case, the normal form $NF_{\mathcal{R}}(B)$ will definitely lie outside $\widehat{\mathcal{A}}$. In other words, normalization of any subterm in B into the set $\widehat{\mathcal{U}}$ is a necessary condition for B to have the truth value "true". So let t be an arbitrary term in H with $NF_{\mathcal{R}}(t) \notin \widehat{\mathcal{U}}$. Then we may in fact substitute t for any subterm u in B with $NF_{\mathcal{R}}(u) \notin \widehat{\mathcal{U}}$ without changing the truth value of B (which is "false"). But then, it is again not necessary to check all possible ground instances $C\vartheta$ of C, that are obtained by substituting all possible terms in H for the variables in C. Instead, it suffices to consider those ground instances $C\vartheta$ of C, where the range of ϑ is restricted to the *finite* set $\widehat{\mathcal{U}} \cup \{t\}$.

Thus, the clause evaluation procedure in [9] for an arbitrary clause C in a GAE-model $\mathcal{M}$ consists of two steps: First, we have to check, whether the whole Herbrand universe H normalizes into $\widehat{\mathcal{U}}$ or not. This is done by the simple procedure CHECK_TOTAL_COVER displayed in Figure 1. Depending on the outcome of this procedure, we have to distinguish two cases:

- *Case 1*: If $\forall t \in H$, $NF_{\mathcal{R}}(t) \in \widehat{\mathcal{U}}$ holds, then the truth value of C can be determined via the following equivalence: C evaluates to "true" in $\mathcal{M}$, iff every ground instance $C\vartheta$ does, where ϑ is an arbitrary ground substitution with range $rg(\vartheta) \subseteq \widehat{\mathcal{U}}$.

CHECK IF $\widehat{\mathcal{U}}$ CONTAINS ALL POSSIBLE NORMAL FORMS
input: Σ, $\mathcal{R}$, $\widehat{\mathcal{U}}$
output: "true", if the whole universe of ground terms (over Σ) normalizes into $\widehat{\mathcal{U}}$;
"t" with $t \in \mathcal{T}(\mathcal{F})$, $NF_{\mathcal{R}}(t) \notin \widehat{\mathcal{U}}$, otherwise.

begin

for all $f \in \Sigma$ with arity $k \geq 0$ **do**
if $\exists f(t_1, \ldots, t_k)$, s.t. $(\forall i \in \{1, \ldots, k\}, t_i \in \widehat{\mathcal{U}})$ **and** $NF_{\mathcal{R}}(f(t_1, \ldots, t_k)) \notin \widehat{\mathcal{U}}$
then return $f(t_1, \ldots, t_k)$;
return "true";

end.

Fig. 1. procedure CHECK_TOTAL_COVER

- *Case 2*: If $\exists t \in H$, s.t. $NF_{\mathcal{R}}(t) \notin \widehat{\mathcal{U}}$ holds, then the truth value of C can be determined via another equivalence, namely: C evaluates to "true" in $\mathcal{M}$, iff every ground instance $C\vartheta$ does, where ϑ is an arbitrary ground substitution with range $rg(\vartheta) \subseteq \widehat{\mathcal{U}} \cup \{t\}$.

Based on these considerations, the correctness of procedure CHECK_TOTAL_COVER is not difficult to verify.

3 Clause Evaluation in GAE-Models

In the previous section, we recalled the clause evaluation procedure from [9]. It is clear "at a glance", that this procedure requires exponential time since, in general, we will have to test exponentially many ground instances of the form $C\vartheta$ of a given clause C. However, no precise complexity analysis of this procedure or of the clause evaluation problem itself is given in [9]. In this section, we shall show that clause evaluation in GAE-models is coNP-complete. Formally, we investigate the following decision problem:

GAE-CLAUSE-EVALUATION
Instance: A tuple $(\Sigma, \mathcal{M}, C)$, where $\mathcal{M}$ is a GAE-model over the universe H with signature Σ and C is a clause over Σ.
Question: Does C evaluate to "true" in $\mathcal{M}$.

Note that for the GAE-CLAUSE-EVALUATION above as well as for the decision problems to be defined in Section 4, we are considering the signature Σ as part of the input. This decision was mainly motivated by the fact that, when we have in mind the model construction process for some clause set $\mathcal{C}$, then the signature is usually not explicitly given either. Instead, it is implicit in the input $\mathcal{C}$. However, it can be easily verified that all the results derived in Sections 3 to 5 hold for any non-trivial signature, also if Σ is not part of the input.

In the following theorem, we show the coNP-completeness of the decision problem GAE-CLAUSE-EVALUATION. As to coNP-membership, it turns out that no new ideas are required. All that is needed is a careful analysis of the algorithm of [9]. The proof of the coNP-hardness works by a straightforward reduction from the 3-SAT problem.

Theorem 1. *The* GAE-CLAUSE-EVALUATION *problem is coNP-complete.*

Proof. As to coNP-membership, we have to show that checking whether a clause C evaluates to "false" in a model $\mathcal{M}$ can be done in non-deterministic polynomial time. To this end, we have a closer look at the algorithm of [9] recalled in Section 2.2: As was already mentioned in Section 2.1, we can compute a canonical GTRS $\mathcal{R}$ as well as the sets $\widehat{\mathcal{A}}$ and $\widehat{\mathcal{U}}$ of atoms and terms, respectively, in normal form w.r.t. $\mathcal{R}$, in polynomial time. Moreover, we claim that also the procedure CHECK_TOTAL_COVER works in polynomial time. This can be seen as follows: For every $f \in \Sigma$ and every term of the form $f(t_1, \dots, t_k)$ with $t_i \in \widehat{\mathcal{U}}$, we have to check, whether $NF_{\mathcal{R}}(f(t_1, \dots, t_k)) \in \widehat{\mathcal{U}}$ or not. Note that, in principle, there are exponentially many terms of the form $f(t_1, \dots, t_k)$, since we are considering the signature Σ as part of the input. However, we only have to search through these terms $f(t_1, \dots, t_k)$ until we finally find one whose normal form w.r.t. $\mathcal{R}$ is outside $\widehat{\mathcal{U}}$. Moreover, all of the arguments t_i thus considered are in normal form w.r.t. $\mathcal{R}$. Hence, for $NF_{\mathcal{R}}(f(t_1, \dots, t_k))$ to be in $\widehat{\mathcal{U}}$, either $f(t_1, \dots, t_k) \in \widehat{\mathcal{U}}$ holds or there exists a rule of the form $f(t_1, \dots, t_k) \to t$ in $\mathcal{R}$, s.t. $t \in \widehat{\mathcal{U}}$. But then, the number of such terms $f(t_1, \dots, t_k)$ that can be normalized into $\widehat{\mathcal{U}}$ is bounded by $|\widehat{\mathcal{U}}| + |\mathcal{R}|$. In other words, the procedure CHECK_TOTAL_COVER has to check at most $|\widehat{\mathcal{U}}| + |\mathcal{R}| + 1$ terms, provided that it halts when the first term with normal form outside $\widehat{\mathcal{U}}$ is detected. Therefore, it only remains to show that for both possible outcomes of the procedure CHECK_TOTAL_COVER, the evaluation of C to "false" can be checked in non-deterministically polynomial time. But this is easy to prove, namely: Suppose that all ground terms of H normalize into $\widehat{\mathcal{U}}$. Then we just have to guess a ground substitution ϑ with $rg(\vartheta) \subseteq \widehat{\mathcal{U}}$ and check in polynomial time that the resulting *ground* clause $C\vartheta$ evaluates to "false" in $\mathcal{M}$. On the other hand, if there exists some $t \in H$ with $NF_{\mathcal{R}}(t) \notin \widehat{\mathcal{U}}$, then we have to guess a ground substitution ϑ with $rg(\vartheta) \subseteq \widehat{\mathcal{U}} \cup \{t\}$ and again check in polynomial time that the resulting *ground* clause $C\vartheta$ evaluates to "false" in $\mathcal{M}$.

In order to prove the coNP-hardness, we reduce the well-known NP-complete problem 3-SAT to the complementary problem of GAE-CLAUSE-EVALUATION: Let $F = (l_{11} \vee l_{12} \vee l_{13}) \wedge \dots \wedge (l_{n1} \vee l_{n2} \vee l_{n3})$ be a Boolean formula with propositional variables in $X = \{x_1, \dots, x_m\}$, s.t. the literals $l_{\alpha\beta}$ in F are either non-negated propositional variables x_i or negated propositional variables $\bar{x}_i$ for some $x_i \in X$. Then we construct the instance $(\Sigma, \mathcal{M}, C)$ of the GAE-CLAUSE-EVALUATION problem, where $\Sigma = \{P, Q, a, b\}$, $C = P(l_{11}, l_{12}, l_{13}) \vee \dots \vee P(l_{n1}, l_{n2}, l_{n3}) \vee Q(x_1, \bar{x}_1) \vee \dots \vee Q(x_m, \bar{x}_m)$, and the GAE-model is given through the atom set $\mathcal{A} = \{P(a, a, a), Q(b, b)\}$ and the empty set $\mathcal{E} = \emptyset$ of equations. Of course, this transformation can be done in polynomial time. Hence, it only remains to show the following equivalence: F is satisfiable $\Leftrightarrow$ C evaluates to "false" in $\mathcal{M}$.

Note that in the above problem definitions, we write x_i and $\bar{x}_i$ both to denote propositional literals in F and first-order variables in C. This dual use of the notation was chosen in order to stress the main idea of the equivalence proof below, namely: The propositional literals x_i and $\bar{x}_i$ will have the truth value "true", iff the first-order variables x_i and $\bar{x}_i$ are instantiated to b. Likewise, the constant term a will be used to encode the truth value "false".

With these preparations it is not difficult to show the above transformation equivalence (cf. [12]). □

4 Model Equivalence

In this section, we investigate the problem of deciding whether two GAE-models $\mathcal{M}_1$ and $\mathcal{M}_2$ over some signature Σ with universe $H = \mathcal{T}(\mathcal{F})$, $\mathcal{F} \subseteq \Sigma$, are equivalent, i.e., whether all H-ground atoms have the same truth value in these two GAE-models. In this case, we write $\mathcal{M}_1 \equiv \mathcal{M}_2$. Actually, it suffices to provide a decision procedure for testing the inclusion of a GAE-model by another one, i.e.: Is every H-ground atom that is "true" in $\mathcal{M}_1$ also "true" in $\mathcal{M}_2$? By abuse of notation, we denote this by $\mathcal{M}_1 \subseteq \mathcal{M}_2$. Of course, $\mathcal{M}_1 \equiv \mathcal{M}_2$ holds iff both $\mathcal{M}_1 \subseteq \mathcal{M}_2$ and $\mathcal{M}_2 \subseteq \mathcal{M}_1$ hold. We thus study the following problems:

GAE-MODEL-EQUIVALENCE
Instance: A tuple $(\Sigma, \mathcal{M}_1, \mathcal{M}_2)$, where $\mathcal{M}_1$ and $\mathcal{M}_2$ are GAE-models over the universe H with signature Σ.
Question: Does $\mathcal{M}_1 \equiv \mathcal{M}_2$ hold?

GAE-MODEL-INCLUSION
Instance: A tuple $(\Sigma, \mathcal{M}_1, \mathcal{M}_2)$, where $\mathcal{M}_1$ and $\mathcal{M}_2$ are GAE-models over the universe H with signature Σ.
Question: Does $\mathcal{M}_1 \subseteq \mathcal{M}_2$ hold?

Recall from Section 2.1, that the GAE-model $\mathcal{M}_1 = (\mathcal{A}, \mathcal{E}_1)$ can efficiently (i.e., in quadratic time) be brought into the form $(\mathcal{P}, \widehat{\mathcal{A}}, \widehat{\mathcal{S}})$, where $\mathcal{P}$ is a canonical GTRS equivalent to $\mathcal{E}_1$, $\widehat{\mathcal{A}}$ is the set of ground atoms $\mathcal{A}$ in normal form, and $\widehat{\mathcal{S}}$ is the set of ground normal forms of all subterms in $\mathcal{A} \cup \mathcal{E}_1$. Likewise, $\mathcal{M}_2 = (\mathcal{B}, \mathcal{E}_2)$ can be transformed into $(\mathcal{R}, \widehat{\mathcal{B}}, \widehat{\mathcal{U}})$. Our GAE-MODEL-INCLUSION algorithm will be based on the construction of the following sets COMP_1 and COMP_2:

Definition 1. (compatible terms) *The sets* $\text{COMP}_1 \subseteq (\widehat{\mathcal{S}} \times \widehat{\mathcal{U}})$ *and* $\text{COMP}_2 \subseteq (\widehat{\mathcal{S}} \times \{\perp\})$ *are defined as follows:*

- $(s, t) \in \text{COMP}_1$*, iff* $\exists r \in H$*, s.t.* $NF_{\mathcal{P}}(r) = s$ *and* $NF_{\mathcal{R}}(r) = t$.
- $(s, \perp) \in \text{COMP}_2$*, iff* $\exists r \in H$*, s.t.* $NF_{\mathcal{P}}(r) = s$ *and* $NF_{\mathcal{R}}(r) \notin \widehat{\mathcal{U}}$.

In other words, we call a term $s \in \widehat{\mathcal{S}}$ "compatible" with a term $t \in \widehat{\mathcal{U}}$, iff s and t can be obtained as the normal forms w.r.t. $\mathcal{P}$ and $\mathcal{R}$, respectively, of some "common predecessor" $r \in H$. Likewise, a term $s \in \widehat{\mathcal{S}}$ is called "compatible" with the auxiliary symbol $\perp$, iff s is the normal form w.r.t. $\mathcal{P}$ of some term r, whereas the normal form of r w.r.t. $\mathcal{R}$ is outside $\widehat{\mathcal{U}}$. In Figure 2, we provide an algorithm for computing these sets COMP_1 and COMP_2.

The idea of the computation of COMP_1 in Step 1 in Figure 2 is as follows: If some term r is contained both in $\widehat{\mathcal{S}}$ and $\widehat{\mathcal{U}}$, then (r, r) is in COMP_1. Now suppose that f is a k-ary function symbol with $k \geq 0$ and (s_1, t_1), $\ldots$, (s_k, t_k) are pairs of compatible terms in COMP_1. Then also $f(s_1, \ldots, s_k)$ and $f(t_1, \ldots, t_k)$ are compatible. Finally, suppose that (s_i, t_i), $1 \leq i \leq k$, have already been found to be compatible and let $s = f(s_1, \ldots, s_k)$, $t = f(t_1, \ldots, t_k)$. Moreover, suppose that $\mathcal{P}$ contains a rule $s \to s'$. Then also $(s', t) \in \text{COMP}_1$ holds. Likewise, if there exists a rule $t \to t'$ in $\mathcal{R}$ then $(s, t') \in \text{COMP}_1$ holds.

Lemma 1. *(computation of* COMP_1*) Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be two GAE-models over some H with signature Σ. Moreover, let $(\mathcal{P}, \widehat{\mathcal{A}}, \widehat{\mathcal{S}})$ and $(\mathcal{R}, \widehat{\mathcal{B}}, \widehat{\mathcal{U}})$ be the normal form representations of $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively, according to Property (d) in Section 2.1. Then STEP 1 of the procedure* COMPUTE_COMP *in Figure 2 computes the set* $\text{COMP}_1 \subseteq \widehat{\mathcal{S}} \times \widehat{\mathcal{U}}$ *of Definition 1 in time $O(|\widehat{\mathcal{S}}| * |\widehat{\mathcal{U}}| * (|\widehat{\mathcal{S}}| + |\mathcal{P}|) * (|\widehat{\mathcal{U}}| + |\mathcal{R}|) * K)$, where K denotes the maximum arity of (function) symbols in Σ.*

Proof. A complete proof of the correctness and of the upper bound on the complexity of STEP 1 of procedure COMPUTE_COMP is given in the full paper [12]. We only sketch the complexity proof here. By construction, $\text{COMP}_1 \subseteq \widehat{\mathcal{S}} \times \widehat{\mathcal{U}}$. Hence, the fixed point is reached after at most $|\widehat{\mathcal{S}} \times \widehat{\mathcal{U}}|$ iterations of the repeat-loop. The (nested) loops "for all $s = f(s_1, \ldots, s_k) \in \widehat{\mathcal{S}}$" and "for all $(f(s_1, \ldots, s_k) \to s) \in \mathcal{P}$" on the one hand, and "for all $t = f(t_1, \ldots, t_k) \in \widehat{\mathcal{U}}$" and "for all $(f(t_1, \ldots, t_k) \to t) \in \mathcal{R}$" on the other hand, come down to touching $(|\widehat{\mathcal{S}}| + |\mathcal{P}|) * (|\widehat{\mathcal{U}}| + |\mathcal{R}|)$ pairs of terms of the form $(f(s_1, \ldots, s_k), f(t_1, \ldots, t_k))$. For each such pair, we have to carry out at most K checks as to whether $(s_i, t_i) \in \text{COMP}_1$ holds. Each such check can be done in constant time, provided that COMP_1 is represented appropriately (e.g., as a matrix $M \in \{0, 1\}^{|\widehat{\mathcal{S}}| \times |\widehat{\mathcal{U}}|}$, s.t. $M_{\alpha\beta} = 1$, iff $(s_\alpha, t_\beta) \in \text{COMP}_1$). The desired upper bound follows easily. □

The computation of COMP_2 in Step 2 of the algorithm in Figure 2 is based on the following idea: Let $s \in \widehat{\mathcal{S}}$ be of the form $s = f(s_1, \ldots, s_k)$. Then there can be basically two reasons why the pair $(s, \perp)$ is in COMP_2: Suppose that for some i we have already shown $(s_i, \perp) \in \text{COMP}_2$. Then, by the subterm closedness of $\widehat{\mathcal{U}}$ (cf. Property (e) in Section 2.1), also $(s, \perp) \in \text{COMP}_2$ holds. On the other hand, suppose that there exist k pairs (s_1, t_1), $\ldots$, $(s_k, t_k) \in \text{COMP}_1$, s.t. $NF_{\mathcal{R}}(f(t_1, \ldots, t_k)) \notin \widehat{\mathcal{U}}$. By the definition of COMP_1 we know that for every $i \in \{1, \ldots, k\}$ the terms s_i and t_i have a common predecessor r_i w.r.t. $\mathcal{P}$ and $\mathcal{R}$, respectively. Hence, also s and $f(t_1, \ldots, t_k)$ have a common predecessor $f(r_1, \ldots, r_k)$ and thus $(s, \perp) \in \text{COMP}_2$. Of course, if some term s is compatible with $\perp$ and $\mathcal{P}$ contains a rule $s \to s'$, then also s' is compatible with $\perp$.

Lemma 2. *(computation of* COMP_2*) Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be two GAE-models over H with signature Σ and let $(\mathcal{P}, \widehat{\mathcal{A}}, \widehat{\mathcal{S}})$ and $(\mathcal{R}, \widehat{\mathcal{B}}, \widehat{\mathcal{U}})$ be as above. Then STEP 2 of*

COMPUTATION OF COMP_1 **AND** COMP_2
input: Σ, $\mathcal{P}$, $\widehat{\mathcal{S}}$, $\mathcal{R}$, $\widehat{\mathcal{U}}$
result: COMP_1 and COMP_2

STEP 1: /* computation of COMP_1 */

$\text{COMP}_1 := \emptyset$;

repeat

for all $s = f(s_1, \ldots, s_k) \in \widehat{\mathcal{S}}$ **do**
  for all $t = f(t_1, \ldots, t_k) \in \widehat{\mathcal{U}}$ **do**
    if $\forall i \in \{1, \ldots, k\}$, $(s_i, t_i) \in \text{COMP}_1$ **then** $\text{COMP}_1 := \text{COMP}_1 \cup \{(s,t)\}$;
  for all $(f(t_1, \ldots, t_k) \to t) \in \mathcal{R}$ **do**
    if $\forall i \in \{1, \ldots, k\}$, $(s_i, t_i) \in \text{COMP}_1$ **then** $\text{COMP}_1 := \text{COMP}_1 \cup \{(s,t)\}$;
od;

for all $(f(s_1, \ldots, s_k) \to s) \in \mathcal{P}$ **do**
  for all $t = f(t_1, \ldots, t_k) \in \widehat{\mathcal{U}}$ **do**
    if $\forall i \in \{1, \ldots, k\}$, $(s_i, t_i) \in \text{COMP}_1$ **then** $\text{COMP}_1 := \text{COMP}_1 \cup \{(s,t)\}$;
  for all $(f(t_1, \ldots, t_k) \to t) \in \mathcal{R}$ **do**
    if $\forall i \in \{1, \ldots, k\}$, $(s_i, t_i) \in \text{COMP}_1$ **then** $\text{COMP}_1 := \text{COMP}_1 \cup \{(s,t)\}$;
od;

until a fixed point is reached;

STEP 2: /* computation of COMP_2 */

$\text{COMP}_2 := \emptyset$;

for all $s = f(s_1, \ldots, s_k) \in \widehat{\mathcal{S}}$ **do**
  if $\exists f(t_1, \ldots, t_k)$, s.t. $(\forall i \in \{1, \ldots, k\}$, $(s_i, t_i) \in \text{COMP}_1)$
    and $NF_{\mathcal{R}}(f(t_1, \ldots, t_k)) \notin \widehat{\mathcal{U}}$ **then** $\text{COMP}_2 := \text{COMP}_2 \cup \{(s, \bot)\}$;

for all $(f(s_1, \ldots, s_k) \to s) \in \mathcal{P}$ **do**
  if $\exists f(t_1, \ldots, t_k)$, s.t. $(\forall i \in \{1, \ldots, k\}$, $(s_i, t_i) \in \text{COMP}_1)$
    and $NF_{\mathcal{R}}(f(t_1, \ldots, t_k)) \notin \widehat{\mathcal{U}}$ **then** $\text{COMP}_2 := \text{COMP}_2 \cup \{(s, \bot)\}$;

repeat

for all $s = f(s_1, \ldots, s_k) \in \widehat{\mathcal{S}}$ **do**
  if $\exists i \in \{1, \ldots, k\}$, $(s_i, \bot) \in \text{COMP}_2$ **then** $\text{COMP}_2 := \text{COMP}_2 \cup \{(s, \bot)\}$;

for all $(f(s_1, \ldots, s_k) \to s) \in \mathcal{P}$ **do**
  if $\exists i \in \{1, \ldots, k\}$, $(s_i, \bot) \in \text{COMP}_2$ **then** $\text{COMP}_2 := \text{COMP}_2 \cup \{(s, \bot)\}$;

until a fixed point is reached;

Fig. 2. procedure COMPUTE_COMP

the procedure COMPUTE_COMP *in Figure 2 computes the set* $\text{COMP}_2 \subseteq \widehat{\mathcal{S}} \times \{\perp\}$ *of Definition 1 in time* $O((|\widehat{\mathcal{S}}| + |\mathcal{P}|) * (|\widehat{\mathcal{U}}| + |\mathcal{R}|) * K)$*, where* K *denotes the maximum arity of (function) symbols in* Σ.

Proof. Again, we only sketch the complexity proof here. For details and for a correctness proof, see [12]. As to the for-loops in front of the repeat-loop, note that the number of iterations of the for-loops is bounded by $|\widehat{\mathcal{S}}|$ and $|\mathcal{P}|$, respectively. We only have to be a bit careful with the checks inside these for-loops: In principle, there are exponentially many terms of the form $f(t_1, \ldots, t_k)$ with $(s_i, t_i) \in \text{COMP}_1$, since we are considering the signature Σ as part of the input. However, we only have to search through these terms $f(t_1, \ldots, t_k)$ until we finally find one whose normal form w.r.t. $\mathcal{R}$ is outside $\widehat{\mathcal{U}}$. In other words, the number of terms $f(t_1, \ldots, t_k)$ thus tested is bounded by 1 plus the number of such terms that can be normalized into $\widehat{\mathcal{U}}$. Recall that by definition of COMP_1, all arguments t_i are in normal form w.r.t. $\mathcal{R}$. Hence, for $NF_{\mathcal{R}}(f(t_1, \ldots, t_k))$ to be in $\widehat{\mathcal{U}}$, either $f(t_1, \ldots, t_k) \in \widehat{\mathcal{U}}$ holds or there exists a rule $f(t_1, \ldots, t_k) \to t$ in $\mathcal{R}$. Thus, the cost of searching for a term $f(t_1, \ldots, t_k)$ with $(s_i, t_i) \in \text{COMP}_1$ and $NF_{\mathcal{R}}(f(t_1, \ldots, t_k)) \notin \widehat{\mathcal{U}}$ is bounded by $O((|\widehat{\mathcal{U}}| + |\mathcal{R}|) * K)$ for each iteration of each of the two for-loops.

Now consider the repeat-loop. By construction, $\text{COMP}_2 \subseteq \widehat{\mathcal{S}} \times \{\perp\}$. Hence, the fixed point is reached after at most $|\widehat{\mathcal{S}}|$ iterations. In the first for-loop, we have to run through the terms $s \in \widehat{\mathcal{S}}$ and check whether one of the arguments of s already occurs in COMP_2. There are at most $|\widehat{\mathcal{S}}| * K$ such checks required. Likewise, in the second for-loop, we have to run through the terms s occurring as left-hand sides of rules in $\mathcal{P}$. Hence, in this case, we end up with an upper bound of $|\mathcal{P}| * K$ on the number of checks. But then the asymptotic complexity of STEP 2 is fully determined by the complexity $O((|\widehat{\mathcal{S}}| + |\mathcal{P}|) * (|\widehat{\mathcal{U}}| + |\mathcal{R}|) * K)$ of the for-loops in front of the repeat-loop. □

We are now ready to solve the GAE-MODEL-INCLUSION problem. The idea of our decision procedure is to search for an atom $P(r_1, \ldots, r_k)$, whose truth value is "true" in $\mathcal{M}_1$ and "false" in $\mathcal{M}_2$. This can be done via the sets of compatible terms in $\text{COMP}_1 \cup \text{COMP}_2$ as follows: Suppose that $P(s_1, \ldots, s_k)$ is in $\widehat{\mathcal{A}}$ but some argument s_i is compatible with $\perp$. Then $P(s_1, \ldots, s_k)$ has a predecessor $P(r_1, \ldots, r_k)$, whose truth value is "true" in $\mathcal{M}_1$ and "false" in $\mathcal{M}_2$. Likewise, suppose that $P(s_1, \ldots, s_k)$ is compatible with an atom $P(t_1, \ldots, t_k)$ (i.e.: all pairs (s_i, t_i) are in COMP_1), s.t. $P(s_1, \ldots, s_k)$ evaluates to "true" in $\mathcal{M}_1$, whereas $P(t_1, \ldots, t_k)$ evaluates to "false" in $\mathcal{M}_2$. Then, by the definition of COMP_1, the atoms $P(s_1, \ldots, s_k)$ and $P(t_1, \ldots, t_k)$ have a common predecessor that is a witness for $\mathcal{M}_1 \not\subseteq \mathcal{M}_2$. Similarly to Lemma 2, it can be shown that such a procedure works in polynomial time. More precisely, we get the following.

Lemma 3. *(*GAE-MODEL-INCLUSION*) Let* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *be two GAE-models over* H *with signature* Σ *and let* $(\mathcal{P}, \widehat{\mathcal{A}}, \widehat{\mathcal{S}})$ *and* $(\mathcal{R}, \widehat{\mathcal{B}}, \widehat{\mathcal{U}})$ *as above. Moreover, suppose that the sets* COMP_1 *and* COMP_2 *of Definition 1 have already been com-*

puted. Then the GAE-MODEL-INCLUSION *problem can be decided in time* $O(|\widehat{\mathcal{A}}| * |\widehat{\mathcal{B}}| * K)$*, where* K *is the maximum arity of (predicate) symbols in* Σ*.*

Proof. By the above considerations, the GAE-MODEL-INCLUSION problem can be solved with the following piece of program code:

```
for all P(s_1, ..., s_k) ∈ Â do
  if ∃i ∈ {1, ..., k}, s.t. (s_i, ⊥) ∈ COMP_2
    then return "false";
  if ∃P(t_1, ..., t_k), s.t. (∀i ∈ {1, ..., k}, (s_i, t_i) ∈ COMP_1)
    and P(t_1, ..., t_k) ∉ B̂ then return "false";
od;
return "true";
```

For the correctness of this procedure, see [12]. As to the complexity, note that the for-loop is iterated at most $|\widehat{\mathcal{A}}|$ times. Moreover, in the first if-statement, we have to check for at most K arguments s_i whether $(s_i, \perp) \in \text{COMP}_2$ holds. Now let us consider the second if-statement. Similarly to Lemma 2, we have to be a bit careful since, in principle, there are exponentially many atoms of the form $P(t_1, \ldots, t_k)$ to be checked. However, we stop this search as soon as we find an atom $P(t_1, \ldots, t_k)$ whose normal form w.r.t. $\mathcal{R}$ is outside $\widehat{\mathcal{B}}$. Hence, analogously to Lemma 2, we get an upper bound $O(|\widehat{\mathcal{B}}| * K)$ on the time required for processing each atom $P(s_1, \ldots, s_k) \in \widehat{\mathcal{A}}$. In total, we thus get the desired upper bound $O(|\widehat{\mathcal{A}}| * |\widehat{\mathcal{B}}| * K)$. □

The main result of this section is now easily established. Recall that by Property (d) in Section 2.1, the computation of $(\mathcal{P}, \widehat{\mathcal{A}}, \widehat{\mathcal{S}})$ from $\mathcal{M}_1$ and of $(\mathcal{R}, \widehat{\mathcal{B}}, \widehat{\mathcal{U}})$ from $\mathcal{M}_2$ can be done in polynomial time, more precisely in quadratic time. Hence, together with the Lemmas 1 through 3, we immediately get

Theorem 2. *The* GAE-MODEL-EQUIVALENCE *problem can be decided in polynomial time, i.e.: Let* $\mathcal{M}_1 = (\mathcal{A}_1, \mathcal{E}_1)$ *and* $\mathcal{M}_2 = (\mathcal{A}_2, \mathcal{E}_2)$ *be two arbitrary GAE-models over some* H *with signature* Σ*. Then it can be decided in polynomial time w.r.t. the input* $(\Sigma, (\mathcal{A}_1, \mathcal{E}_1), (\mathcal{A}_2, \mathcal{E}_2))$ *whether* $\mathcal{M}_1 \equiv \mathcal{M}_2$ *holds.*

Remark (Comparison with other methods): An alternative way of solving the equivalence problem of GAE-models would be to transform the GAE-models into finite tree automata (FTA, for short) and to apply an equivalence test to the resulting FTA. In [14, 15], such a transformation from the normal form representation $(\mathcal{P}, \widehat{\mathcal{A}}, \widehat{\mathcal{S}})$ of a GAE-model $\mathcal{M}_1$ into an equivalent deterministic FTA $A_1 = (\Sigma, Q, Q_f, \delta)$ (i.e.: A_1 recognizes exactly those ground atoms over Σ which are "true" in $\mathcal{M}_1$) is presented. The number $|Q|$ of states of A_1 basically corresponds to the number $|\widehat{\mathcal{S}}|$ of normalized subterms in the GAE-model. Moreover, the number of state transitions in δ basically corresponds to $|\mathcal{P}| + |\widehat{\mathcal{A}}|$.

Now suppose that we have transformed both GAE-models $\mathcal{M}_1$ and $\mathcal{M}_2$ into equivalent FTA A_1 and A_2. Let N denote the total number of states of these automata and let M denote their total size. With the construction of [14, 15], N depends linearly on $|\widehat{\mathcal{S}}| + |\widehat{\mathcal{U}}|$ and M depends linearly on the total size of $(\mathcal{P}, \widehat{\mathcal{A}}, \widehat{\mathcal{S}})$ and $(\mathcal{R}, \widehat{\mathcal{B}}, \widehat{\mathcal{U}})$. In order to test the equivalence of these two FTA, we could follow the way suggested in [6], i.e., first transform the FTA into equivalent *complete* FTA A_1' and A_2'. Then the equivalence of A_1' and A_2' can be checked in quadratic time. However, recall that we are considering the signature as part of a problem instance here. Hence, the straightforward process of completing an FTA may add exponentially many new transition rules in the worst-case.

Alternatively, we may test the equivalence of A_1 and A_2 via the method presented in [18]. A careful analysis of [18] reveals that in our setting this method works in time $O(N^{K+2} * M)$, where K denotes the maximum arity of the symbols in Σ. Moreover, it is shown in [18] how the maximum arity K can be reduced to 2. Note however, that this reduction of the arity leads to a multiplication of the number of states, i.e., we get automata A_1' and A_2' whose total number of states is basically $N * K$. Hence, we end up with an upper bound $O((N * K)^4 * M)$ on the time complexity.

The main source of complexity of our model equivalence test is the computation of the sets COMP_1 and COMP_2. By Lemma 1, this can be done in time $O(|\widehat{\mathcal{S}}| * |\widehat{\mathcal{U}}| * (|\widehat{\mathcal{S}}| + |\mathcal{P}|) * (|\widehat{\mathcal{U}}| + |\mathcal{R}|) * K)$. In terms of the FTA, this corresponds to $O(N^2 * M^2 * K)$. The computations according to the Lemmas 2 and 3 can also be done within this bound. Actually, if the original GAE-model contains a huge number of equations on comparatively few and small terms, then the algorithm of [18] may possibly be more efficient than ours. Otherwise (i.e., in particular, when the number of subterms occurring in the GAE-model is of the same order of magnitude as the number of equations) our algorithm should actually outperform the one hidden in [18]. Moreover, the dependence on the signature is clearly smaller in our algorithm.

5 Construction of an Almost Minimal Finite Model

In [9], an algorithm is provided that constructs a GAE-model $\mathcal{M}$ for any satisfiable clause set from the so-called class $\text{PVD}_g^=$. Of course, if the signature Σ contains at least one proper function symbol, then in general the model $\mathcal{M}$ has an infinite domain, namely the Herbrand universe over Σ. However, it is shown in [9], how one can derive from $\mathcal{M}$ an equivalent finite model $\mathcal{M}_{\mathcal{F}}$. This extraction of a finite model is based on the following idea: First of all, the model $\mathcal{M}$ given through a set of ground atoms $\mathcal{A}$ and ground equations $\mathcal{E}$ is transformed into the form $(\mathcal{R}, \widehat{\mathcal{A}}, \widehat{\mathcal{U}})$, where $\mathcal{R}$ is a canonical GTRS equivalent to $\mathcal{E}$, $\widehat{\mathcal{A}}$ contains the normal forms of the atoms in $\mathcal{A}$, and $\widehat{\mathcal{U}} = \{t_1, \ldots, t_N\}$ contains the normal forms of all terms and subterms occurring in $\mathcal{A}$ and $\mathcal{E}$. Then, analogously to the clause evaluation procedure recalled in Section 2.2, we first have to check, whether the whole universe H over the signature Σ normalizes into $\widehat{\mathcal{U}}$ or not.

This can be done with the procedure CHECK_TOTAL_COVER in Figure 1. Then we have to distinguish two cases:

- Case 1: If $\forall t \in H$, $NF_{\mathcal{R}}(t) \in \widehat{\mathcal{U}}$, then $\mathcal{M}_{\mathcal{F}}$ has the domain $D = \{a_1, \ldots, a_N\}$ (i.e., D has the same cardinality as $\widehat{\mathcal{U}}$), s.t. every ground term $s \in H$ with normal form $NF_{\mathcal{R}}(s) = t_i$ is interpreted by a_i. Moreover, each predicate symbol $P \in \Sigma$ with arity $k \geq 1$ is interpreted in $\mathcal{M}_{\mathcal{F}}$ by the relation $\mathcal{M}_{\mathcal{F}}(P) \subseteq D^k$ with $\mathcal{M}_{\mathcal{F}}(P) = \{(a_{i_1}, \ldots, a_{i_k}) \,|\, P(t_{i_1}, \ldots, t_{i_k}) \in \widehat{\mathcal{A}}\}$.
- Case 2: If $\exists t \in H$ with $NF_{\mathcal{R}}(t) \notin \widehat{\mathcal{U}}$, then $\mathcal{M}_{\mathcal{F}}$ has the domain $D = \{a_1, \ldots, a_N\} \cup \{d\}$ for some new element d (i.e., $|D| = |\widehat{\mathcal{U}}| + 1$), s.t. every ground term $s \in H$ with normal form $NF_{\mathcal{R}}(s) = t_i \in \widehat{\mathcal{U}}$ is interpreted by a_i and every ground term $s \in H$ with normal form $NF_{\mathcal{R}}(s) \notin \widehat{\mathcal{U}}$ is interpreted by d. Analogously to Case 1 above, each predicate symbol $P \in \Sigma$ with arity $k \geq 1$ is interpreted by the relation $\mathcal{M}_{\mathcal{F}}(P) = \{(a_{i_1}, \ldots, a_{i_k}) \,|\, P(t_{i_1}, \ldots, t_{i_k}) \in \widehat{\mathcal{A}}\}$.

Recall from Section 3, that the computational cost of evaluating clauses over a GAE-model $\mathcal{M}$ heavily depends on the size of $\widehat{\mathcal{U}}$. In particular, if C is a clause with k variables, then we have to check $k^{|\widehat{\mathcal{U}}|}$ ground instances of C (in Case 1 above) or $k^{(|\widehat{\mathcal{U}}|+1)}$ instances (in Case 2 above), respectively, in order to show that C indeed evaluates to "true" in $\mathcal{M}$. Likewise, if we do the clause evaluation in the corresponding finite model, then the computational cost is mainly determined by the size of the finite domain D. Consequently, before we actually start evaluating clauses in a GAE-model $\mathcal{M}$ (e.g., when $\mathcal{M}$ is used by theorem prover based on semantic resolution), it would be desirable to first transform $\mathcal{M}$ into an equivalent GAE-model $\mathcal{M}'$ with a smaller set $\widehat{\mathcal{U}}'$. Likewise, if the clause evaluation is done in the finite model $\mathcal{M}_{\mathcal{F}}$ obtained from $\mathcal{M}$ via the algorithm of [9], then we should first search for an equivalent finite model $\mathcal{M}'_{\mathcal{F}}$ with a smaller domain.

Actually, searching for a finite model whose domain is minimal by checking all possible domains D with $|D| = 1, 2, 3, \ldots$ and all possible interpretations of the signature Σ is out of the question. It was already pointed out in [9] that constructing a GAE-model first and extracting a finite model afterwards is the only reasonable way. However, it was also mentioned in [9] that this method has the disadvantage that the size of the domain of the resulting finite model is possibly far bigger than actually necessary. In this section, we show how this drawback can be remedied to a certain extent. Of course, it is infeasible to start from scratch and compute a completely different model. However, what we can do is to take a GAE-model $\mathcal{M}$ resulting from the model construction procedure in [9] and transform it into an equivalent model $\mathcal{M}'$ with a possibly much smaller set $\widehat{\mathcal{U}}'$ of terms in normal form. Consequently, also the finite model $\mathcal{M}'_{\mathcal{F}}$ obtained from $\mathcal{M}'$ by the above recalled method of [9] possibly has a much smaller domain than the finite model $\mathcal{M}_{\mathcal{F}}$ extracted from $\mathcal{M}$. In Figure 3 we provide an algorithm that takes a GAE-model $\mathcal{M}$ as an input and transforms it into an equivalent model $\mathcal{M}'$ that is "almost minimal" in the following sense.

Definition 2. (almost minimal GAE-model) *Let $\mathcal{M}$ be a GAE-model and let $\mathcal{M}_{\mathcal{F}}$ with domain D be the corresponding finite model that is obtained via the algorithm of [9]. Furthermore, let $\mathcal{M}_{\min}$ be an equivalent finite model with minimal domain $D_{\min}$. Then we call $\mathcal{M}$ a* minimal *GAE-model, if $|D| = |D_{\min}|$ holds. Moreover, $\mathcal{M}$ is called an* almost minimal *GAE-model if $|D| \leq |D_{\min}| + 1$ holds.*

```
COMPUTATION OF AN ALMOST MINIMAL GAE-MODEL
input: Σ, M = (A, E)
output: M' = (A', E'), s.t. M' is almost minimal and equivalent to M
begin
  compute (R, Â, Û) according to (A, E);    /* cf. Property (d) in Section 2.1 */
  repeat
    if ∃ distinct terms s, t in Û, s.t.
      M = (A, E) and M' = (A, E ∪ {s = t}) are equivalent then
    begin
      E := E ∪ {s = t};
      compute (R, Â, Û) according to (A, E);
    end;
  until no such pair (s, t) is found;
  return (A, E);
end.
```

Fig. 3. procedure FIND_ALMOST_MINIMAL_MODEL

The procedure FIND_ALMOST_MINIMAL_MODEL in Figure 3 is based on the following idea: Let $\mathcal{M}$ be given in the form $(\mathcal{R}, \widehat{\mathcal{A}}, \widehat{\mathcal{U}})$. Now suppose that all terms in $\widehat{\mathcal{U}}$ are interpreted in $\mathcal{M}_{\min}$ by pairwise distinct elements of $D_{\min}$. In this case, we have $|\widehat{\mathcal{U}}| \leq |D_{\min}|$ and, therefore, $|D| \leq |D_{\min}| + 1$. Hence, $\mathcal{M}$ is already almost minimal. On the other hand, suppose that there are two distinct terms s and t in $\widehat{\mathcal{U}}$, that are interpreted by the same domain element $a_i \in D_{\min}$. In this case, it is easy to show that adding the equation $s = t$ to the GAE-model yields an equivalent model. Hence, in order to transform $\mathcal{M}$ into an almost minimal GAE-model we just have to take all pairs (s, t) from $\widehat{\mathcal{U}}$ and check whether the equation $s = t$ can be added to the GAE-model without changing the truth value of any ground atom (of $\mathcal{M}$) over this signature.[9]

[9] Note that, in practice and from an efficiency point of view, an incremental completion in procedure FIND_ALMOST_MINIMAL_MODEL is of course preferable. That is, instead of completing $\mathcal{E} \cup \{s = t\}$ it is generally better to use the already computed canonical GTRS $\mathcal{R}$ for $\mathcal{E}$ and complete $\mathcal{R} \cup \{s = t\}$ (using the same completion parameters, in particular, the same reduction ordering).

Theorem 3. *(termination and correctness of* FIND_ALMOST_MINIMAL_MODEL*)* *Let $\mathcal{M} = (\mathcal{A}, \mathcal{E})$ be a GAE-model over the signature Σ. Then the procedure* FIND_ALMOST_MINIMAL_MODEL *with input $(\Sigma, \mathcal{M})$ terminates and returns an almost minimal GAE-model $\mathcal{M}'$ that is equivalent to $\mathcal{M}$.*

Proof: As to termination, recall from Property (e) in Section 2.1, that $\widehat{\mathcal{U}}$ is closed w.r.t. subterms. By construction, the terms in $\widehat{\mathcal{U}}$ correspond to equivalence classes of all subterms occurring in $\mathcal{A}$ and $\mathcal{E}$ w.r.t. the equations in $\mathcal{E}$. Of course, we get exactly the same set of equivalence classes w.r.t. the equations in $\mathcal{E}$, if we consider the subterms of $\mathcal{A}$ and $\mathcal{E}$ plus the subterms of $s, t \in \widehat{\mathcal{U}}$. Moreover, adding an equation $s = t$ to $\mathcal{E}$ with $s, t \in \widehat{\mathcal{U}}$, $s \neq t$, has the following effect: On the one hand, the two distinct equivalence classes corresponding to s and t collapse into a single one and, on the other hand, no new equivalence class can arise by adding an equation to $\mathcal{E}$. Hence, the number of equivalence classes w.r.t. $\mathcal{E} \cup \{s = t\}$ of the subterms in $\mathcal{A}$ and $\mathcal{E} \cup \{s = t\}$ is strictly smaller than the number of equivalence classes w.r.t. $\mathcal{E}$ of the subterms in $\mathcal{A}$ and $\mathcal{E}$. But then, the number of iterations of the repeat-loop is clearly restricted by $|\widehat{\mathcal{U}}|$ of the input GAE-model $\mathcal{M}$.

As to correctness of the procedure FIND_ALMOST_MINIMAL_MODEL, we have to show that the resulting GAE-model $\mathcal{M}'$ is almost minimal and equivalent to $\mathcal{M}$. Actually, the equivalence is trivial since, in each iteration of the repeat-loop, we only add an equation $s = t$ to $\mathcal{E}$ after checking that the GAE-models $(\mathcal{A}, \mathcal{E})$ and $(\mathcal{A}, \mathcal{E} \cup \{s = t\})$ are indeed equivalent. It only remains to show that the final model $\mathcal{M}'$ is almost minimal. To this end, it suffices to show that $|\widehat{\mathcal{U}}'| \leq |D_{\min}|$ holds. In other words, any two distinct terms in $\widehat{\mathcal{U}}'$ are interpreted by distinct domain elements in $D_{\min}$. Suppose on the contrary, that $\widehat{\mathcal{U}}'$ contains two distinct terms s, t, which are interpreted by the same domain element in $D_{\min}$. Now let $r[s]$ be an arbitrary ground term with subterm s and let $r[t]$ be the term obtained from $r[s]$ by replacing the subterm s with t. Then the interpretation $\mathcal{M}_{\min}$ maps the terms $r[s]$ and $r[t]$ to identical domain elements in $D_{\min}$. Likewise, if $A[s]$ is a ground atom with subterm s and if $A[t]$ is obtained from $A[s]$ by replacing the subterm s with t, then the interpretation $\mathcal{M}_{\min}$ clearly assigns the same truth value to $A[s]$ and $A[t]$. In other words, ground atoms that are equivalent w.r.t. the equation $s = t$ are assigned the same truth value by the interpretation $\mathcal{M}_{\mathcal{F}}$. Hence, they are also assigned the same truth value by $\mathcal{M}'$. Thus we may in fact add the equation $s = t$ to $\mathcal{E}'$ without changing the model $\mathcal{M}'$. But this contradicts the assumption, that $\mathcal{M}'$ is the final result of procedure FIND_ALMOST_MINIMAL_MODEL. □

The algorithm FIND_ALMOST_MINIMAL_MODEL is put to work in the following example:

Example 2. Let $\mathcal{M} = (\mathcal{A}, \mathcal{E})$ be a GAE-model over the signature $\Sigma = \{a, b, f\}$ with $\mathcal{A} = \{P(f^2(a)),\ P(a)\}$ and $\mathcal{E} = \{f^2(a) = f^4(a), f^2(b) = a\}$. Clearly, $\mathcal{U} = \{a, f(a), f^2(a), f^3(a),\ f^4(a), b, f(b), f^2(b),\ f^3(b), f^4(b)\}$. Moreover, completing $\mathcal{E}$ into a canonical GTRS $\mathcal{R}$ and normalizing $\mathcal{T}$ yields $\mathcal{R} = \{f^4(a) \to$

$f^2(a), f^2(b) \to a\}$,[10] $\widehat{\mathcal{U}} = \{a, f(a), f^2(a), f^3(a), b, f(b)\}$. In the repeat-loop of procedure FIND_ALMOST_MINIMAL_MODEL, we search for a pair $s = t$ of terms in $\widehat{\mathcal{U}}$, s.t. $(\mathcal{A}, \mathcal{E})$ and $(\mathcal{A}, \mathcal{E} \cup \{s = t\})$ are equivalent: We first check the equation $a = f(a)$. Actually, it can be easily checked "by hand" that $(\mathcal{A}, \mathcal{E})$ and $(\mathcal{A}, \mathcal{E} \cup \{s = t\})$ are not equivalent. This is due to the fact that $P(a)$ and $P(f(a))$ clearly have the same truth values in $(\mathcal{A}, \mathcal{E} \cup \{s = t\})$, but different ones in $\mathcal{M}$. Now we check the equation $a = f^2(a)$. In fact, this equation may be added to $\mathcal{E}$. Hence, we have to update the variables $\mathcal{E}$, $\mathcal{R}$, and $\widehat{\mathcal{U}}$ in the procedure FIND_ALMOST_MINIMAL_MODEL as follows:

$$\mathcal{E} = \{f^2(a) = f^4(a), f^2(b) = a, f^2(a) = a\},$$
$$\mathcal{R} = \{f^2(a) \to a, f^2(b) \to a\},$$
$$\widehat{\mathcal{U}} = \{a, f(a), b, f(b)\}.$$

In the next iteration of the repeat-loop, we check whether the equation $a = b$ may be added to the new set $\mathcal{E}$. The answer is again yes. Hence, the new values of the variables $\mathcal{E}$, $\mathcal{R}$, and $\widehat{\mathcal{U}}$ are:

$$\mathcal{E} = \{f^2(a) = f^4(a), f^2(b) = a, f^2(a) = a, b = a\},$$
$$\mathcal{R} = \{f^2(a) \to a, b \to a\},$$
$$\widehat{\mathcal{U}} = \{a, f(a)\}.$$

We have already mentioned above, that the equation $a = f(a)$ must not be added. Hence, the repeat-loop terminates and returns the final result $\mathcal{M}' = (\mathcal{A}', \mathcal{E}')$ with $\mathcal{A}' = \mathcal{A}$ and $\mathcal{E}' = \{f^2(a) = f^4(a), f^2(b) = a, f^2(a) = a, b = a\}$.

6 Discussion

The aim of this paper was to improve upon previous algorithms for the clause evaluation problem and the model equivalence problem of GAE-models. As to model equivalence, we have presented a polynomial time algorithm whereas all previously published methods require exponential time. Yet, as we have pointed out in Section 4, a careful analysis of previous work and results on tree automata and of their relation to GAE-models (cf. [18], [6], [14, 15]) has revealed that an appropriate combination of such known results also yields indeed a polynomial time algorithm for the model equivalence problem of GAE-models. A comparison of this "hidden" generic approach with our special purpose algorithm seems to indicate that ours should usually outperform the former one.

As to clause evaluation we have established a clear limit for possible improvements by proving the coNP-completeness. Nevertheless, the computation of an "almost minimal" GAE-model presented in Section 5 may lead to a considerable speed-up in many cases. It has already been mentioned that our notion of GAE-models slightly differs from the definition in [9] in that we consider (ground) equality only as a representation formalism but we do not consider (ground) equations as (ground) atoms. Actually, if we consider the notion of GAE-models

[10] For the construction of $\mathcal{R}$, we assume here that $b > a$ holds, w.r.t. to the underlying reduction ordering used.

from [9] then all of the results in Sections 3 and 4 should easily carry over to this case. The transformation of GAE-models in Section 5 does not become wrong but it gets useless. Namely, if we consider ground equations as atoms that may occur in the clauses, then we will never be allowed to add any equations (as in Fig. 3) to the set $\mathcal{E}$ without changing the model.[11]

Let us finally compare our results with algorithms and complexity results for a few other model representation formalisms. The complexity behaviour of linear or general atomic representations (i.e., LARs and ARs, for short) of Herbrand models is different from GAE-models. In [8], ARs are defined as finite sets of atoms $\mathcal{A} = \{A_1, \dots, A_n\}$ with the intended meaning that the ground atoms that evaluate to "true" in such a model are precisely the ground instances of the atoms in $\mathcal{A}$. In an LAR, each atom $A_i \in \mathcal{A}$ has to be linear (i.e.: A_i contains no multiple occurrences of variables). It can be shown that LARs have strictly less expressive power than GAE-models while ARs are incomparable with GAE-models (cf. [15]). In contrast to GAE-models, both the clause evaluation problem *and* the model equivalence problem have been shown to be coNP-complete for LARs and ARs (cf. [11]).

Apart from the model representation formalisms mentioned above, there are many more in the literature. In [15] (and partly also in [14]) a comparison of quite a few such formalisms is given from the point of view of their expressive power. However, rather little attention has been paid so far to algorithmic and complexity-theoretic aspects of most of these formalisms, although the construction of efficient algorithms is decisive for their practical value. Hence, a lot of work remains to be done this area.

Acknowledgement: We would like to thank the anonymous referees for a couple of useful hints and constructive criticisms.

References

1. F. Baader and T. Nipkow. *Term rewriting and All That.* Cambridge University Press, 1998.
2. P. Baumgartner, C. Fermüller, N. Peltier, and H. Zhang. Workshop: Model Computation – Principles, Algorithms, Applications. In D. McAllester, ed., *Proc. 17th Int. Conf. on Automated Deduction (CADE'00)*, LNAI 1831, p. 513, Pittsburgh, PA, USA, June 2000. Springer-Verlag.
3. R. Caferra and N. Peltier. Extending semantic resolution via automated model building: Applications. In *Proc. 14th Int. Conf. on Artificial Intelligence (IJCAI'95)*, pp. 328–334, Montréal, Québec, Canada, Aug. 1995. Morgan Kaufmann.
4. R. Caferra and N. Zabel. A method for simultanous search for refutations and models by equational constraint solving. *Journal of Symbolic Computation*, 13:613–642, 1992.
5. H. Chu and D. Plaisted. CLIN-S - a semantically guided first-order theorem prover. *Journal of Automated Reasoning*, 18(2):183–188, 1997.

[11] In this setting, identification of distinct ground term congruence classes clearly destroys (at least the equational part of) the original model.

6. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Preliminary version from October, 14 1999, available at: http://www.grappa.univ-lille3.fr/tata.
7. N. Dershowitz and D. Plaisted. Rewriting. In J. Robinson and A. Voronkov, eds., *Handbook of Automated Reasoning*, volume 1, chapter 9, pp. 535–610. Elsevier and MIT Press, 2001.
8. C. G. Fermüller and A. Leitsch. Hyperresolution and automated model building. *Journal of Logic and Computation*, 6(2):173–230, 1996.
9. C. G. Fermüller and A. Leitsch. Decision procedures and model building in equational clause logic. *Logic Journal of the IGPL*, 6(1):17–41, 1998.
10. Z. Fülöp and S. Vágvölgyi. Ground term rewriting rules for the word problem of ground term equations. *Bulletin of the European Association for Theoretical Computer Science*, 45:186–201, Oct. 1991.
11. G. Gottlob and R. Pichler. Hypergraphs in model checking: Acyclicity and hypertree-width versus clique-width. In F. Orejas, P. Spirakis, and J. Leeuwen, eds., *Proc. 28th International Colloquium on Automata, Languages and Programming (ICALP'01)*, LNCS 2076, pp. 708–719, Crete, Greece, July 201. Springer-Verlag.
12. B. Gramlich and R. Pichler. Algorithmic aspects of Herbrand models represented by ground atoms with ground equations. Technical report, Institut für Computersprachen, TU Wien, May 2002. Full version of this paper.
13. K. Hodgson and J. Slaney. System description: Scott-5. In R. Goré, A. Leitsch, and T. Nipkow, eds., *Proc. 1st Int. Joint Conf. on Automated Reasoning (IJCAR'01)*, LNAI 2083, pp. 443–447, Siena, Italy, June 2001. Springer-Verlag.
14. R. Matzinger. Comparing computational representations of Herbrand models. In G. Gottlob, A. Leitsch, and D. Mundici, eds., *Proc. 5th Kurt Gödel Colloquium – Computational Logic and Proof Theory (KGC'97)*, LNCS 1289, pp. 203–218, Vienna, Austria, Aug. 1997. Springer-Verlag.
15. R. Matzinger. *Computational Representations of Models in First-Order Logic*. PhD thesis, Vienna University of Technology, 2000.
16. N. Peltier. Tree automata and automated model building. *Fundamenta Informaticae*, 30(1):59–81, 1997.
17. D. Plaisted and A. Sattler-Klein. Proof lengths for equational completion. *Information and Computation*, 125(2):154–170, 1996.
18. H. Seidl. Deciding equivalence of finite tree automata. *SIAM Journal on Computing,*, 19(3):424–437, June 1990.
19. J. Slaney. FINDER: Finite domain enumerator – system description. In A. Bundy, ed., *Proc. 12th Int. Conf. on Automated Deduction (CADE'94)*, LNAI 814, pp. 798–801, Nancy, France, June 26 – July 1 1994. Springer-Verlag.
20. W. Snyder. A fast algorithm for generating reduced ground rewriting systems from a set of ground equations. *Journal of Symbolic Computation*, 15:415–450, 1993.
21. T. Tammet. *Resolution Methods for Decision Problems and Finite Model Building*. PhD thesis, Chalmers University of Technology, Göteborg, Sweden, 1992.
22. S. Winker. Generation and verification of finite models and counterexamples using an automated theorem prover answering two open questions. *Journal of the ACM*, 29(2):273–284, 1982.
23. J. Zhang and H. Zhang. System description: Generating models by SEM. In M. McRobbie and J. Slaney, eds., *Proc. 13th Int. Conf. on Automated Deduction (CADE'96)*, LNAI 1104, pp. 308–312, New Brunswick, NJ, USA, July 30 – August 3 1996. Springer-Verlag.