Hyperresolution and Automated Model Building

CHRISTIAN FERMÜLLER and ALEXANDER LEITSCH, *Technische Universität Wien, 1040 Vienna, Austria.*
E-mail: {*chrisf, leitsch*} *@logic.tuwien.at*

Abstract

Building on previous results that show that hyperresolution refinements may usefully be employed as decision procedures for a wide range of decidable classes of clause sets we present methods for constructing models for such sets of clauses. We show how to generate finite sets of atoms that respresent Herbrand models using hyperresolution. We demonstrate that these atomic representations of models enjoy features that provide a basis for various applications in automated theorem proving. In particular, we show that the equivalence of atomic representations is decidable and that arbitrary clauses can be evaluated effectively w.r.t. to the represented models. For the investigated classes we may focus on atoms with a linear term structure and show that, for this important subcase, finite models can be extracted from the sets of atoms generated by hyperresolution. We emphasize that, in contrast to model theoretic approaches, no backtracking is needed in our proof theoretic model constructing algorithm.

Keywords: Resolution, decision procedures, model building.

1 Introduction

Finding and investigating models of abstract structures is at the very heart of mathematical activity. Consequently the value of using models in automated deduction is widely admitted. But although mathematical logic has uncovered an impressive body of knowledge about models of first-order formulas, very little is known about algorithmical methods for model building. In the area of first-order prefix classes, most of the research was directed to prove the *existence* of models, without giving explicit *representations* of models. In particular, the existence of finite models for satisfiable formulas of a class Δ (such classes are called finitely controllable) implies the decidability of Δ. Thus showing finite controllability of classes is a key proof technique in the theory of decidable classes [4]. A characteristic of this technique is to compute a recursive bound $\alpha(F)$ for every F in Δ s.t. F is satisfiable iff there exists a model of domain size $\leq \alpha(F)$. The transformation of such a proof into an algorithmical method results in exhaustive search through all finite domain interpretations of size $\leq \alpha(F)$. But, even for small $\alpha(F)$, this is clearly inadequate for practical computing. In more recent time algorithmical finite model building based on theorem proving methods has been investigated by T. Tammet [21, 22], Manthey and Bry [14], Caferra and Zabel [1, 2] and J. Slaney [19, 20]. An earlier approach of S. Winker [23], although practically relevant and successful, did not define a general algorithmic method.

Tammet's approach, like ours, is based on resolution decision procedures. But his finite model building method applies to the monadic and Ackermann class only and is based on the termination of an ordering refinement. In the resulting model description the interpretation of the function symbols is given completely, but the interpretation of predicate symbols is only partial. Moreover, Tammet uses narrowing and works with equations on the object language level. In this paper the model building is based on termination sets for hyperresolution (which yield other decision classes). The finite model building method of Section 5 is based on the transformation of Herbrand models; it does not use equality reasoning but filtration.

Caffera and Zabel define an (equational) extension of the resolution calculus, specifically designed for the purpose of finite model construction. They give complete representations of Herbrand models but do not specify finite models completely (only the interpretation of the predicate symbols is given). Their method is not based on 'postprocessing' of termination sets of resolution but considerably alters the inference system itself. Moreover, the decidable classes are completely different from ours.

Manthey and Bry describe a hyperresolution prover based on a model generation paradigm in [14]. Their method of model building, although similar in the case of the class PVD_+, differs from ours in several aspects. They essentially use splitting of positive ground clauses and backtracking, features that are avoided in our approach. Moreover, we can handle cases where no ground facts are produced. While Manthey and Bry construct Herbrand models only, the final products of our procedure are finite models.

Slaney [19] devised the program *FINDER* that identifies finite models (of reasonably small cardinality) of clause sets whenever they exist. The algorithm is a clever variant of exhaustive search through all possible interpretations and is not referring to resolution or any other inference system. In [20] Slaney investigates the advantages of combining the resolution-based theorem prover *OTTER* [16] with *FINDER*. Although the methods underlying this approach are quite different form ours its main point—demonstrating the usefulness of model building in automated deduction—serves well as motivation also for our work.

We present a fully algorithmical method for the construction of models out of termination sets of positive hyperresolution provers. Even more importantly, we develop a formal framework for model building by resolution based on resolution operators (including condensation and subsumption) and orthogonalization. Our key concept is that of a (finite) atomic representation of a Herbrand model. We argue that such atomic representations are suitable and useful as model descriptions by demonstrating that it is decidable whether two given atomic representations are equivalent and that arbitrary clauses can effectively be evaluated w.r.t. the represented models. Motivated by special syntactical properties of the the investigated examples of decidable classes we focus on linear atomic representations, i.e. sets of atoms in which a variable occurs at most once. In particular, we show that linear atomic respresentations can be 'orthogonalized'. This fact is exploited in order to project the corresponding Herbrand models into finite models.

The basic idea of our approach to automated model building is the following: If the (refutationally complete) theorem prover stops on a set of clauses $\mathcal{C}$ without deriving $\Box$ (contradiction) then $\mathcal{C}$ must be satisfiable. Having obtained the (finite) set $\mathcal{D}$ of derivable clauses, we start to construct a model out of $\mathcal{D}$ (i.e. we use the information provided by positive hyperresolution). The first step of model building consists in transforming the set of derived positive clauses $\mathcal{D}_+$ into an a finite set of unit clauses $\mathcal{U}$ by iteratively applying hyperresolution operators. $\mathcal{U}$ is shown to be an atomic representation of a Herbrand model of $\mathcal{C}$. In general, this model is infinite since its domain is the Herbrand universe. The second step is of a different flavour. As already mentioned above, we present a method which, for a wide range of classes decidable by hyperresolution, constructs finite models out of the atomic representations generated by hyperresolution. Neither backtracking nor equality reasoning on the object level is required (the latter in contrast to [21, 22, 1], the former in contrast to [14, 19]).

Although the finite models obtained in this way are not minimal (w.r.t. domain size) in general, the method is clearly superior to exhaustive search, since no kind of backtracking is involved. However, it is limited to syntax classes admitting filtration on the termination sets (i.e. Herbrand models can be mapped into finite models under preservation of truth values).

As the method presented in this paper is based on termination of positive hyperresolution with subsumption and condensing, the preprocessing step just consists of 'ordinary' theorem proving. Only in case $\Box$ has not been derived, the proper model building algorithm is applied to the termination set. This is also a characteristic of Tammet's work, while Caferra and Zabel start with a model building procedure at once. Together with Tammet's results on finite model building, this paper can be considered as a starting point for model building methods based on resolution decision procedures for first-order classes.

Some results of this paper were already introduced in [6]. However, there are numerous new concepts and results like the notion of H-subsumption, the decidability of equivalence of atomic representations and the effective evaluation of clauses w.r.t. the represented models. Moreover, the transformation of termination sets into atomic representations is treated more exactly.

2 Basic notions

We assume the reader to be familiar with clause logic and the concept of resolution. In particular, *terms*, *atoms*, and *literals* are defined as usual. By an *expression* we mean a term or a literal. A *clause* is a finite disjunction of literals. The *empty clause* (representing contradiction) is denoted by $\Box$. The *size of a clause* C, i.e. the number of literals occurring in C, is denoted by $|C|$.

We consider clauses to be in some normal form. More exactly, the *or*-connective is treated modulo associativity; we may assume a fixed, e.g. lexicographic, ordering of the literals in a clause and that multiple occurrences of the same literal are removed. (We still prefer to conceive of clauses as disjunctions and not as sets, since this allows more elegant formulations of some of our results.) Moreover, we implicitly assume the names of the variables to be determined by their order of occurrence within a clause. Thus, for example, $P(x, y) \vee Q(y, z)$ and $P(w, u) \vee Q(u, v)$ denote the same clause. Observe that these conventions imply that, for a finite alphabet of predicate, constant and function symbols, there are only finitely many different clauses of fixed depth and size.

We write $C - L$ for the subclause of C obtained by deleting the literal L in C.

By a *clause set* we mean a finite set of clauses.

$V(E)$ denotes the set of variables occurring in an expression, clause or clause set E. An expression or clause E is called *ground* if $V(E) = \emptyset$. A clause C is called *decomposed* if $V(L) \cap V(M) = \emptyset$ for all literals L and M occurring in C.

To be able to argue concisely we need some additional notions:

DEFINITION 2.1
If C is a clause then C_+ is the set of positive (unsigned) literals in C, analogously C_- denotes the set of negative literals (negated atoms) in C. C is a called *Horn* if it contains at most one positive literal, i.e. if $|C_+| \leq 1$. C is *positive* if $C_- = \emptyset$. For a set of clauses $\mathcal{C}$ $P(\mathcal{C})$ denotes the subset of positive clauses in $\mathcal{C}$ and $NP(\mathcal{C}) = \mathcal{C} - P(\mathcal{C})$, the subset of *non-positive* clauses.

It is convenient to make use of some definitions concerning term structure that are well known from the term rewriting literature:

DEFINITION 2.2
By $\mathcal{P}(A)$ we denote the set of all *positions*, considered as finite sequences of integers, of an expression A [9]. $[A|p]$ denotes the subterm of A at position p; $\langle A|p\rangle$ denotes the symbol occurring in A at position p.

p is called a *subposition* of p' if p is an initial sequence of p' (possibly identical to p').

The *length of a position* $|p|$ is the number of integers in p.

The *number of occurrences* of a variable x in an expression A is defined as $OCC(x, A) = |\{p \mid p \in \mathcal{P}(A) \text{ and } [A|p] = x\}|$. The definition is extended to clauses and sets of expressions in an obvious manner.

The following notion is known from [15]:

DEFINITION 2.3
The *frontier* of any two expressions A_1 and A_2, $FRONT(A_1, A_2)$, is the set of pairs of terms which occur at the same position in A_1 and A_2, respectively, but have a different leading symbol. More formally $FRONT(A_1, A_2) = \{([A_1|p], [A_2|p]) \mid \langle A_1|p\rangle \neq \langle A_1|p\rangle \text{ and } p \in \mathcal{P}(A_1) \cap \mathcal{P}(A_2)\}$.

DEFINITION 2.4
The term depth of an expression A is defined as $\tau(A) = \max\{|p| \mid p \in \mathcal{P}(A)\}$.

The *maximal depth of occurrence* of a variable x in $V(A)$ is defined as $\tau_{\max}(x, A) = \max\{|p| \mid p \in \mathcal{P}(A) \text{ and } [A|p] = x\}$. Similarly, $\tau_{\min}(x, A) = \min\{|p| \mid p \in \mathcal{P}(A) \text{ and } [A|p] = \{x\}$. Again, these definitions are generalized to clauses and sets of expressions in the obvious way.

DEFINITION 2.5
Let V be the set of variables and T be the set of terms. A *substitution* is a mapping $\sigma : V \to T$ s.t. $\sigma(x) = x$ almost everywhere. We call the set $dom(\sigma) = \{x|\sigma(x) \neq x\}$ *domain* of σ and $rg(\sigma) = \{\sigma(x)|x \in dom(\sigma)\}$ *range* of σ.

A substitution σ is called *ground* if all terms in $rg(\sigma)$ are ground. It is said to be *based on* a set of terms H or shortly an H-substitution if $rg(\sigma) \subseteq H$. The set of all H-substitutions is denoted by $GS(H)$.

If E is an expression or clause and σ is an (H-)substitution then the result of applying σ to E is denoted by $E\sigma$. $E\sigma$ is called an *(H)-instance* of E. An instance E_1 of E_2 is called *proper* if $E_1 \neq E_2$ (up to a renaming of variables).

The set of all ground H-instances of elements of a set of expressions T is denoted by $G_H(T)$.

REMARK 2.6
Whenever we speak of *the* set of ground substitutions or instances we implicitly refer to some fixed Herbrand universe on which the substitutions are assumed to be based.

Most general unifiers (mgus) of sets of expressions and *(binary) resolvents* and *factors* of clauses are defined as usual [3].

DEFINITION 2.7
Let E_1 and E_2 be expressions, then $E_1 \leq_{sub} E_2$—read: E_1 *is more general than* E_2—iff there exists a substitution σ s.t. $E_1\sigma = E_2$.

Similary, if C and D are clauses, $C \leq_{sub} D$ if there exists a substitution σ s.t. $C\sigma$ is a subclause of D. In this case we say that C *subsumes* D.

We extend the subsumption relation to sets of clauses $\mathcal{C}, \mathcal{D}$ by defining $\mathcal{C} \leq_{sub} \mathcal{D}$ if for all $D \in \mathcal{D}$ there exists a $C \in \mathcal{C}$ s.t. $C \leq_{sub} D$. We write $\mathcal{C} =_{sub} \mathcal{D}$ if both $\mathcal{C} \leq_{sub} \mathcal{D}$ and $\mathcal{D} \leq_{sub} \mathcal{C}$.

By $sub(\mathcal{C})$ we denote a subset of a clause set $\mathcal{C}$ s.t. for no pair of different clauses $C, D \in sub(\mathcal{C})$ it holds $C \leq_{sub} D$.

REMARK 2.8
Of course, $sub(\mathcal{C})$ is not unique in general, but it can be made so by using specific selection strategies. Since we are not interested in the particular form of the clauses but only make use of the fact that any two different clauses in $sub(\mathcal{C})$ do not subsume each other, we can conceive of sub as a function.

DEFINITION 2.9
A smallest subclause C' of C s.t. $C \leq_{sub} C'$ is called a *condensation* of C. For a set of clauses $\mathcal{C}$ $cond(\mathcal{C})$ denotes the set of condensations of all clauses in $\mathcal{C}$. If C and D are clauses whose condensations are equal (up to a renaming of variables) then we write $C =_c D$.

REMARK 2.10
Remember that condensations are unique up to a renaming of variables if clauses are defined as sets of literals [10]. Below, we shall assume clauses to be in condensed normal form, i.e. we only consider sets of clauses $\mathcal{C}$ where $\mathcal{C} = cond(\mathcal{C})$.

For any finite set of constant and function symbols S, the set of all ground terms built up from symbols in S is called a *Herbrand universe* (over S). For any set of expressions or clause set $\mathcal{C}$ the Herbrand universe of $\mathcal{C}$, $H(\mathcal{C})$, is the Herbrand universe over the constant and function symbols occurring in $\mathcal{C}$. The Herbrand base, $\mathbf{HB}(\mathcal{C})$, is the set of all atoms with predicate symbols occurring in $\mathcal{C}$ and terms from $H(\mathcal{C})$ as arguments.

An *interpretation* $\mathcal{I}$ is denoted as triple $\langle D, \varphi, d\rangle$, where D is the domain, φ the signature interpretation, and d the variable assignment of $\mathcal{I}$. Each interpretation $\mathcal{I} = \langle D, \varphi, d\rangle$ induces an evaluation function $v_{\mathcal{I}}$ in the usual way s.t. $v_{\mathcal{I}}(t) \in D$ for terms t and $v_{\mathcal{I}}(C) \in \{\mathbf{true}, \mathbf{false}\}$ for all literals and clauses C. A *model* for a set of clauses $\mathcal{C}$ is an interpretation $\mathcal{I}$ s.t. $v_{\mathcal{I}}(C) = \mathbf{true}$ for all $C \in \mathcal{C}$.

REMARK 2.11
A clause corresponds to the universal closure of the disjunction of its literals. This means that the truth value of a clause under an interpretation does not depend on the variable assignment. Since we are only interested in the truth values of clauses and ground atoms we may omit variable assigments from the definition of interpretations.

DEFINITION 2.12
A *Herbrand interpretation* of a clause set $\mathcal{C}$ is an interpretation $\mathcal{I} = \langle D, \varphi\rangle$ s.t. $D = H(\mathcal{C})$ and $\varphi(c) = c$ for all constants and $\varphi(f)(t_1, \dots, t_n) = f(t_1, \dots, t_n)$ for all n-ary function symbols and all ground terms t_i. A Herbrand interpretation that verifies all clauses in $\mathcal{C}$ is a *Herbrand model* of $\mathcal{C}$

Herbrand interpretations are uniquely determined by sets of ground atoms and a reference to some alphabet. (We usually drop the reference to an alphabet since, when speaking of a model of some clause set $\mathcal{C}$, we always refer implicitly to the set of predicate, constant and function symbols occurring in $\mathcal{C}$.) We may therefore identify Herbrand interpretations $\mathcal{I}_H$ with subsets of the Herbrand base. The intended meaning is that, for any ground atom $A = P(t_1, \dots t_n)$: $A \in \mathcal{I}_H$ iff $(t_1, \dots, t_n) \in \varphi_{\mathcal{I}_H}(P)$. It follows that $\mathcal{I}_H \subseteq \mathbf{HB}(\mathcal{C})$ is a Herbrand model for a clause set $\mathcal{C}$ iff for all ground instances $C\gamma$ of a clause $C \in \mathcal{C}$ there is either a positive literal A in $C\gamma$ s.t. A is in $\mathcal{I}_H$ or a negative literal $\neg A$ in $C\gamma$ s.t. $A \notin \mathcal{I}_H$.

3 Hyperresolution as decision procedure

All model building operations presented in this paper are based on (positive) hyperresolution. This resolution refinement is used first in a preproccessing manner, in the sense that the model construction itself starts on finite termination sets, i.e. finite sets of clauses that are satisfiable and closed w.r.t. the resolution operator. But also the computation of atomic representations out of the termination sets essentially employs hyperresolution. Termination sets are familiar from resolution decision theory [7]. In this section we formally define hyperresolution in terms of resolution operators. Moreover we present some classes of clause sets that can be decided by hyperresolution. These classes serve as concrete examples for our model building procedures, to be decribed in Sections 5 and 8.

DEFINITION 3.1
Let C, D be condensed clauses, where D is positive. The condensation of a binary resolvent of C and a factor of D is called a *PRF-resolvent.* (PRF abbreviates 'positive, restricted factoring'.)

REMARK 3.2
Throughout this paper we assume that clauses always appear in condensed form, mostly without mentioning this fact explicitly.

DEFINITION 3.3
Let C be a non-positive clause and let the clauses D_i, for $1 \leq i \leq n$, be positive. Then the sequence $\Gamma = (C; D_1, \ldots, D_n)$ is called a *clash sequence.*

Let $C_0 = C$ and C_{i+1} be a PRF-resolvent of C_i and D_{i+1} for $i < n$. If C_n is positive then it is called a *clash resolvent* (or *hyper resolvent*) defined by Γ.

Hyperresolution examplifies the principle of macro inference. It only produces positive clauses or the empty clause $\Box$. In variance to the standard definition of hyperresolution we have included a restriction on factoring. The concept of 'semi-factoring' is investigated in [17], where—inter alia—it is shown that positive hyperresolution based on PRF-resolution is complete.

Below, we do not need to refer to hyperresolution deductions themselves but rather are interested in the set of derived clauses. For this purpose the following operator-based description of hyperresolution seems most adequate.

DEFINITION 3.4
Let $\mathcal{C}$ be a set of clauses. By $\rho_H(\mathcal{C})$ we denote the set of all clash resolvents definable by clash sequences of clauses in $\mathcal{C}$. The hyperresolution operator R_H and its closure R_H^* is defined by:

$$R_H(\mathcal{C}) = \mathcal{C} \cup \rho_H(\mathcal{C}).$$

$$R_H^0(\mathcal{C}) = \mathcal{C} \text{ and } R_H^{i+1}(\mathcal{C}) = R_H(R_H^i(\mathcal{C})) \text{ for } i \leq 0.$$

$$R_H^*(\mathcal{C}) = \bigcup_{i \geq 0} R_H^i(\mathcal{C}).$$

Combining hyperresolution with subsumption we obtain:

DEFINITION 3.5
Let $\mathcal{C}$ be a set of clauses and *sub* be the subsumption reduction operator as defined in Section 2. Then we have

$$R_{HS}(\mathcal{C}) = sub(\mathcal{C} \cup \rho_H(\mathcal{C})).$$

$R^0_{HS}(\mathcal{C}) = sub(\mathcal{C})$ and $R^{i+1}_{HS}(\mathcal{C}) = R_{HS}(R^i_{HS}(\mathcal{C}))$ for $i \leq 0$.

$$R^*_{HS}(\mathcal{C}) = \bigcap_{i \geq 0} \bigcup_{j \geq i} R^j_H(\mathcal{C}).$$

REMARK 3.6
In contrast to resolution operators without deletion mechanisms, like R_H, $R^*_{HS}(\mathcal{C})$ is not simply defined as $\bigcup_{i\geq 0} R^i_{HS}(\mathcal{C})$. The reason is that R_{HS} is not monotonic, i.e. $\mathcal{C} \subseteq R_{HS}(\mathcal{C})$ does not hold in general. Thus R^*_{HS} is defined as 'limes superior', and not as union, of the $R^i_{HS}(\mathcal{C})$.

R^*_{HS} enjoys two important properties:

(1) R_{HS} is complete, i.e. $\square \in R^*_{HS}(\mathcal{C})$ whenever $\mathcal{C}$ is unsatisfiable. Note that, by definition of the subsumption operator, $\square \in R^*_{HS}(\mathcal{C})$ implies $R^*_{HS}(\mathcal{C}) = \{\square\}$. (Since $\square$ subsumes all clauses.)

(2) If there exsists an i s.t. $R^i_{HS}(\mathcal{C}) \leq_{sub} R^{i+1}_{HS}(\mathcal{C})$, i.e. if $R^i_{HS}(\mathcal{C})$ is a fixed point w.r.t. R_{HS}, then $R^*_{HS}(\mathcal{C}) = R^i_{HS}(\mathcal{C})$ and $R^*_{HS}(\mathcal{C})$ is finite. (Of course, $\mathcal{C}$ is assumed to be finite.)

It is well known that hyperresolution remains complete when combined with subsumption [13]. The crucial fact here is that $R^*_{HS}(\mathcal{C}) \leq_{sub} R^*_H(\mathcal{C})$. That condensation preserves completeness is shown in [7].

If for all $\mathcal{C}$ in a class of clause sets Γ there exists an i s.t. $R^*_{HS}(\mathcal{C}) = R^i_{HS}(\mathcal{C})$ then the computation of the finite fixed point $R^*_{HS}(\mathcal{C})$ defines a decision procedure for Γ; we say that R_{HS} *decides* Γ. Note that $R^*_{HS}(\mathcal{C}) \subseteq R_H(\mathcal{C})$, and therefore R_{HS} decides Γ whenever R_H decides Γ.

We now define two classes which can be decided by R_{HS}.

DEFINITION 3.7
PVD$_+$ is the set of all sets of clauses $\mathcal{C}$ s.t. for all $C \in \mathcal{C}$:

(1) $V(C_+) \subseteq V(C_-)$, and
(2) for all $x \in V(C_+)$: $\tau_{\max}(x, C_+) \leq \tau_{\max}(x, C_-)$.

(Observe, that (1) implies that all positive clauses in $\mathcal{C}$ must be ground.)

PVD$_+$ is a subclass of PVD, a class that has been demonstrated to be decidable in [12] and [7]. But there is no loss of generality in investigating PVD$_+$ instead of PVD, since the decision procedure for PVD can be easily reduced to that for PVD$_+$: A clause set $\mathcal{C}$ is in PVD iff there is a renaming η of the signs of the literals s.t. $\eta(\mathcal{C}) \in$ PVD$_+$. To obtain a resolution decision procedure for PVD we first construct an adequate renaming η and then apply R_{HS} to $\eta(\mathcal{C})$. Instead of transforming $\mathcal{C}$ to $\eta(\mathcal{C})$ one may also apply a different semantical setting [13] to $\mathcal{C}$ itself.

PVD$_+$ can be considered as a generalization of DATALOG to clause forms containing function symbols. The decidability of PVD$_+$ restricted to Horn was shown in [11].

DEFINITION 3.8
OCC1N$_+$ is the set of all sets of clauses $\mathcal{C}$ s.t. for all $C \in \mathcal{C}$:

(1) $OCC(x, C_+) = 1$ for all $x \in V(C_+)$, and
(2) $\tau_{\max}(x, C_+) \leq \tau_{\min}(x, C_-)$ for all $x \in V(C_+) \cap V(C_-)$.

Like in the case of PVD_+, $\mathcal{C} \in \mathrm{OCC1N}$ iff there exists a renaming η of the signs of the literals s.t. $\eta(\mathcal{C}) \in \mathrm{OCC1N}_+$. OCC1N is shown to be decidable in [7]; the decidability of $\mathrm{OCC1N}_+$ restricted to Horn clauses is proven in [5].

PVD_+ is even decidable by positive hyperresolution without condensing [12]. However, in deciding $\mathrm{OCC1N}_+$ condensing is essential (but subsumption is not needed, see [7]). Clearly, if a resolution procedure terminates (i.e. if only finitely many resolvents are produced) then this remains true if condensing and subsumption is added. Therefore the cited results imply that R_{HS} decides $\mathrm{PVD}_+ \cup \mathrm{OCC1N}_+$. We shall see that the reduction of the clause sets w.r.t. condensing and subsumption is essential to the model building procedure.

In the following sections the sets $R^*_{HS}(\mathcal{C})$ will serve as 'raw material' for model building. All results obtained in this paper hold for the decidable classes PVD_+ and $\mathrm{OCC1N}_+$, but many of them are more general. A wider range of decidable classes (generalizations of PVD) was obtained in [12] using the concept of abstract measures of atom complexity. All these classes are decidable by hyperresolution and admit the finite model building procedure introduced in Section 8.

4 Representing Herbrand models

Observe that speaking of algorithms that construct models of clause sets is actually a slight misuse of language. The output of a computer program can, of course, only be a formal description or *representation* of a model. As long as we only deal with finite models there is hardly a point in insisting on the difference between the model (as abstract mathematical entity) and its representation (e.g. in the form of multiplication tables). However, in dealing with infinite structures the difference gets essential. Here we are interested in Herbrand models (which are infinite in all but trivial cases) and thus have to present a formal language for the representation of these structures. It should be clear that there is no single formalism capable of representing *all* Herbrand models (over a signature that contains at least one function symbol) in a finite manner, since there are non-denumerably many.

Our aim is to specify an algorithm that, given a satisfiable clause set $\mathcal{C}$ in a class that is decidable by hyperresolution as indicated in Section 3, constructs a representation of *some* Herbrand model of $\mathcal{C}$. We shall show that these models allow for a particularly simple and elegant representation in the form of finite sets of atoms.

DEFINITION 4.1
Let $\mathcal{A}$ be a finite set of atoms and H be a Herbrand universe containing $H(\mathcal{A})$. We say that $\mathcal{A}$ is an *atomic representation* of the set $INT_H(\mathcal{A})$ of all H-instances of $\mathcal{A}$. Remember that we identify Herbrand models with the set of true ground atoms; therefore $\mathcal{A}$ may be considered as an atomic representation of the Herbrand model $INT_H(\mathcal{A})$. We abbreviate this by saying that $\mathcal{A}$ is an ARM (w.r.t. to H).

Observe that whenever only finitely many ground atoms are true in a Herbrand model $\mathcal{I}$ then $\mathcal{I}$ (as a set of atoms) and its atomic representation coincide. However, many more interesting Herbrand models admit atomic representations as well.

EXAMPLE 4.2
Let $\mathcal{C} = \{P(f(a)),\ P(x) \vee P(f(x)),\ \neg P(a)\}$. Then $\mathcal{I} = \{P(f(t)) \mid t \in H(\mathcal{C})\}$ is a Herbrand model of $\mathcal{C}$. The set $\mathcal{A} = \{P(f(x))\}$ is obviously an atomic representation of $\mathcal{I}$ w.r.t. $H(\mathcal{C})$, i.e. $\mathcal{I} = INT_{H(\mathcal{C})}(\mathcal{A})$.

Of course, there are (even quite simple) Herbrand models that do not have an atomic representation. For example, if we augment $\mathcal{C}$ by the clause $\neg P(x) \vee \neg P(f(x))$ then $\mathcal{I}' = \{P(f^{(2n+1)}(a)) \mid n \geq 0\}$ is the only Herbrand model of the new clause set. But $\mathcal{I}'$ cannot be represented by a finite number of atoms in the sense of Definition 4.1.

Whether a formalism is adequate for the representation of models not only depends on which type of models we want to describe, but also on the intended applications. However, even if we abstract from specific applications there are some features which seem desirable for any type of representation formalism:

- It should be (computationally) easy to decide whether a given ground atom is true or false in the represented model, (e.g. an algorithm that evaluates ground atoms in polynomial time w.r.t. the size of the atom and the representation of the model would be adequate).
- It should be possible to decide whether two given representations specify the same model or not.
- There should be an effective method for the evaluation of arbitrary clauses w.r.t. the represented model, i.e. there should be an algorithm that decides whether a given clause is true or false in the model.

Observe that in the case of finite models the standard representation by multiplication tables almost trivially fulfills these requirements of adequacy. However, as soon as we want to represent infinite models it is not clear at all how to evaluate clauses or how to test the equivalence of representations. For atomic representations of Herbrand models it is clear that arbitrary ground atoms can be evaluated efficiently: Since a ground atom B is true in $INT_H(\mathcal{A})$ iff there is some $A \in \mathcal{A}$ s.t. B is an instance of A, the evaluation consists in a linear number of instance checks and thus is at most of quadratic time complexity. In Sections 6 and 7 we shall demonstrate that the other features from the above list also hold. We do not know of any other representation mechanism for reasonable classes of infinite models that shares these desirable properties.

Observe, that the possibility to evaluate effectively arbitrary clauses is a basic requirement if one wants to use model constructing algorithms to find counter models during proof search (e.g. in the frame of model elimination procedures.) Recently, there is a lot of interest in this type of model-based search pruning [20], which is usually employed in a purely heuristic, ad hoc manner and still seems to lack a theoretical foundation. Our results also contribute to this line of research. Another application for model representation mechanisms that allow for the evaluation of clauses (and literals) consists in the possibility to use more sophisticated interpretations than simple 'settings' in the context of semantic resolution [13].

The decidable classes introduced in the last section not only allow for the construction of atomic representation of models but also guarantee a very simple term structure of the representing atoms. The resulting representations are called linear.

DEFINITION 4.3
An expression E is called *linear* if each variable in E occurs only once, i.e. $OCC(x, E) = 1$ for all $x \in V(E)$. A set of expressions is said to be linear if all its elements are linear.

We show that every linear representation can algorithmically be transformed into a finite set of atoms that represents the same Herbrand model in an 'orthogonal' manner and consists of instances of the original atoms. We shall use such orthogonal representations in Section 8 to project the represented Herbrand models into finite models.

W.l.o.g. we speak of terms only in the following definitions. The generalization to atoms is obvious.

DEFINITION 4.4
Let T be a finite set of non-variable linear terms and H a Herbrand universe containing $H(T)$. T *orthogonally represents* the set of its ground H-instances $G_H(T)$ if for all $s \in G_H(T)$ there is exactly one $t \in T$ s.t. s is an instance of t.

T is an *orthogonal extension* of a set of terms T' (w.r.t. H) if T orthogonally represents $G_H(T')(= G_H(T))$ and if for each $t' \in T'$ there is an H-instance t of t' in T.

EXAMPLE 4.5
$T = \{a,\ f(x, f(u,v)),\ f(a,y),\ f(x,a)\}$ represents $G_H(T)$, where H is the set of all terms built up from a and f only. Indeed, $G_H(T) = H$. T is not an orthogonal representation since, for example, $f(a, f(a,a))$ is an instance of both, $f(x, f(u,v))$ and $f(a,y)$. $T_1 = \{a, f(x,y)\}$ orthogonally represents H but it is not an orthogonal extension of T since, for example, no instance of $f(a,y)$ occurs in T_1. However, $T_2 = \{a,\ f(x, f(u,v)),\ f(a,a),\ f(f(x,y),a)\}$ is an orthogonal extension of T w.r.t. H.

REMARK 4.6
Observe that T is an orthogonal representation iff any two different terms in T are not unifiable.

Below we present an algorithm that constructs for any finite set T of linear terms an orthogonal extension of T. This algorithm is based on a technique to achieve mutual disunification of terms. It should be noted that methods of disunification are also basic to the model building calculus in [1] and [2]. There, the conditions of disunification form an important part of the equational constraint language.

DEFINITION 4.7
The set of *linear base terms* BT_H of some Herbrand universe H is the set of all constants in H and all linear terms of the form $f(x_1, \ldots, x_n)$, where f is a function symbol in the alphabet on which H is based.

REMARK 4.8
Within a set of terms we do not distinguish terms that only differ in the names of their variables. Therefore BT_H contains exactly one term for each constant and function symbol. It can be considered as the simplest orthogonal representation of H. (Remember that a variable cannot be an element of a representation by definition.)

We need to represent the set of all terms that are not instances of a certain term:

DEFINITION 4.9
Let t be a linear term and H a Herbrand universe. Then the set of terms $co_H(t)$ is inductively defined as follows:

(1) If t is a constant then $co_H(t) = BT_H - \{t\}$.
(2) If $t = f(t_1, \ldots, t_n)$ then $co_H(t) = (BT_H - \{f(x_1, \ldots, x_n)\}) \cup \{f(s_1, \ldots, s_n) \mid s_i = t_i$ or $s_i \in co_H(t_i)$ but at least one $s_i \neq t_i\}$, where $V(s_i) \cap V(s_j) = \emptyset$ for all $i \neq j$; i.e. the variable names are chosen s.t. $f(s_1, \ldots, s_n)$ is linear. (Observe that $co_H(t)$ is undefined for variables; thus if t_i is a variable then $s_i = t_i$.)

The relativation of $co_H(t)$ to a term s, $co(t|s)$, is defined as the subset of $co_H(t)$ that contains all terms in $co_H(t)$ that are instances of s.

Observe that all elements of $co_H(t)$ are linear terms.

EXAMPLE 4.10
Let $t = f(x,a)$, and let H be the set of ground terms built up from a, f, and the unary function symbol g. Then $co_H(a) = \{g(x),\ f(x,y)\}$ and therefore $co_H(t) = \{a,\ g(x),\ f(x,g(y)),\ f(x,f(y,z))\}$. $co_H(t|f(x,y)) = \{f(x,g(y)),\ f(x,f(y,z))\}$.

The following lemmas serve to guarantee the correctness of our orthogonalization algorithm.

LEMMA 4.11
For any linear term t $co_H(t)$ is an orthogonal representation of $H - G_H(\{t\})$, i.e. the set of all terms of H that are not instances of t.

PROOF. Consider the following propositions:

(1) $\{t\} \cup co_H(t)$ represents H.
(2) No $s \in co_H(t)$ is unifiable with t.
(3) $\forall s_1, s_2 \in co_H(t)$: s_1 is not unifiable with s_2 unless $s_1 = s_2$.

(1) and (2) together imply that $co_H(t)$ represents $H - G_H(\{t\})$, i.e. the set of all terms in H that are not instances of t. (3) implies that the representation is orthogonal. We prove (1), (2), and (3) simultaneously by induction on the term depth of t:

Obviously, the propositions hold if t is a constant. We have the following induction hypothesis:

(IH) Propositions (1), (2), and (3) hold for all t where $\tau(t) \leq n$.

Let $t' = f(t_1, \ldots, t_k)$ be any term of depth $n+1$. Since by the induction hypothesis $\{t_i\} \cup co_H(t_i)$ represents H for each $1 \leq i \leq k$, it follows by part (2) of Definition 4.9 that $co_H(t')$ represents $H - G_H(t')$. This proves proposition (1).

Clearly, no base term except $f(x_1, \ldots, x_n)$ is unifiable with t'. On the other hand for any term $f(s_1, \ldots, s_n) \in co_H(t')$ there is at least one i with $s_i \in co_H(t_i)$. It follows by part (2) of (IH) that also $f(s_1, \ldots, s_n)$ is not unifiable with t'. This proves proposition (2).

To prove proposition (3) first observe that any two different base terms are not unifiable. Let $s = f(s_1, \ldots, s_n)$ and $s' = f(s'_1, \ldots, s'_n)$ be two different elements of $co_H(t')$. By definition, there must be an i s.t. $s_i \neq s'_i$ and both terms are not variables. Hence, by part (ii) of Definition 4.9, for some i either both, s_i and s'_i, are in $co_H(t_i)$ or exactly one of the two terms equals t_i. In either case, the non-unifiability of s and s' follows by the induction hypothesis. ∎

LEMMA 4.12
For any H-instance $t\theta$ of a linear term t, $co_H(t\theta|t)$ is an orthogonal represention of $G_H(\{t\}) - G_H(\{t\theta\})$, i.e. the set of all ground H-instances of t that are not instances of $t\theta$.

PROOF. $co_H(t\theta|t)$ is defined as the subset of all terms in $co_H(t\theta)$ that are instances of t. We first consider the set $co'_H(t\theta|t)$ of all terms in $co_H(t\theta)$ that are unifiable with t, and then show that $co'_H(t\theta|t) = co_H(t\theta|t)$.

Since $co'_H(t\theta|t)$ is a subset of $co_H(t\theta)$ it follows from Lemma 4.11 that the subset $co'_H(t\theta|t)$ is orthogonal and does not represent any instances of $t\theta$, i.e. $G_H(co'_H(t\theta|t)) \subseteq G_H(\{t\}) - G_H(\{t\theta\})$. To see that the converse also holds consider any term $s \in G_H(\{t\}) - G_H(\{t\theta\})$. Since s is not in $G_H(\{t\theta\})$ there must be some $s' \in co_H(t\theta)$ s.t. s is an instance of s'. But s is also an instance of t by definition. Consequently, t and s' are unifiable and thus $s' \in$

$co'_H(t\theta|t)$. This means that $s \in G_H(co'_H(t\theta|t))$. Therefore $G_H(co'_H(t\theta|t)) = G_H(\{t\}) - G_H(\{t\theta\})$.

It remains to show that any term $s' \in co'_H(t\theta|t)$ is also an instance of t. Suppose to the contrary that there is a position $p \in \mathcal{P}(s') \cap \mathcal{P}(t)$ s.t. $[s'|p]$ is a variable but $[t|p]$ is not a variable. (If there is no such p then, since s' and t are unifiable, s' is an instance of t.) By the definition of $co_H(t\theta)$, the fact that $[s'|p]$ is a variable implies that there is some subposition p' of p in $\mathcal{P}(t\theta)$ s.t. $[t\theta|p']$ is a variable, too. Since $t\theta$ is an instance of t we conclude that there exists a subposition p'' of p' in $\mathcal{P}(t)$ s.t. $[t|p'']$ is a variable. But this clearly contradicts the assumption that $[t|p]$ is not a variable. We have thus shown that s' is an instance of t and consequently $co'_H(t\theta|t) = co_H(t\theta|t)$.

Summarizing we have proved that $co_H(t\theta|t)$ is an orthogonal represention of $G_H(\{t\}) - G_H(\{t\theta\})$. ∎

We are now prepared to present an orthogonalization algorithm.

```
function Orthogonalize
{Input: finite set $T$ of linear terms, alphabet of $H$}
{Output: orthogonal extension of $T$ w.r.t. $H$}
begin
    $T' := T$;
    while $\exists t_1, t_2 \in T' (t_1 \neq t_2)$ s.t. $t_1$ and $t_2$ are unifiable do
        begin
        $\theta :=$ mgu of $t_1$ and $t_2$;
        $T' := (T' - \{t_1, t_2\}) \cup \{t_1\theta (= t_2\theta)\} \cup co_H(t_1\theta|t_1) \cup co_H(t_1\theta|t_2)$
        end { while };
    return $T'$
end.
```

Remarks. H is assumed to be based on an alphabet that at least contains the constant and function symbols occurring in T. Moreover, it should be clear that the terms are considered to be variable disjoint when subjected to the unification algorithm. (Thus, for example, $f(x, g(y))$ is unifiable with $f(g(x), y) (= f(g(u), v))$.)

Before proving the correctness of *Orthogonalize* we illustrate its behaviour by an example.

EXAMPLE 4.13
Let $T = \{f(x, g(y)),\ g(x),\ g(f(x, a)),\ f(x, a),\ a,\ f(g(x), y)\}$ and H be based on the alphabet consisting of a, g and f, only. Denoting T as

$$T = \{t_1, t_2, \ldots t_6\}$$

where

$$t_1 = f(x, g(y)),\ t_2 = g(x),\ t_3 = g(f(x, a)),\ t_4 = f(z, a),\ t_5 = a,\ t_6 = f(g(u), v),$$

one easily sees that (t_1, t_6), (t_2, t_3) and (t_4, t_6) are pairs of unifiable terms.

We apply *Orthogonalize* to T and chose t_1 and t_6 as terms to be replaced in $T = T'^{(0)}$. The substitution $\sigma = \{x \leftarrow g(u),\ v \leftarrow g(y)\}$ is an mgu of (t_1, t_6). Thus $t_1\sigma = t_6\sigma = f(g(u), g(y))$. We have

$$co_H(t_1\sigma|t_1) = \{f(a, g(x))(= t_7),\ f(f(x, y), g(z))(= t_8)\} \text{ and}$$

$$co_H(t_1\sigma(=t_6\sigma)|t_6) = \{f(g(x),a)(=t_9),\ f(g(x),f(y,z))(=t_{10})\}.$$

We arrive at

$$T'^{(1)} = \{t_2,t_3,t_4,t_5,t_7,t_8,t_9,t_{10},t_{11} = t_1\sigma(=t_6\sigma)\}.$$

In $T'^{(1)}$ $t_4 = f(z,a)$ and $t_9 = f(g(x),a))$ are unifiable with mgu $\sigma = \{z \leftarrow g(x)\}$. $t_4\sigma(= t_9\sigma = t_9) = f(g(x),a)$, thus

$$co_H(t_4\sigma|t_4) = \{f(a,a) = t_{12},\ f(f(x,y),a)(=t_{13})\} \text{ and}$$

$$co_H(t_4\sigma(=t_9)|t_9) = \emptyset.$$

We obtain

$$T'^{(2)} = \{t_2,t_3,t_5,t_7,t_8,t_9,t_{10},t_{11},t_{12},t_{13}\},$$

$t_2 = g(x)$ and $t_3 = g(f(x,a)) = g(f(y,a))$ are unifiable with mgu $\sigma = \{x \leftarrow f(y,a)\}$.

$$\begin{aligned} co_H(t_2\sigma|t_2) = \{&g(a)(=t_{14}),\ g(g(x))(=t_{15}), g(f(x,g(y))(=t_{16}),\\ &g(f(x,f(y,z))(=t_{17})\} \qquad \text{and}\end{aligned}$$

$$co_H(t_2\sigma(=t_3)|t_3) = \emptyset.$$

We therefore have

$$T'^{(3)} = \{t_3,t_5,t_7,t_8,t_9,t_{10},t_{11},t_{12},t_{13},t_{14},t_{15},t_{16},t_{17},\}.$$

Since no pair of different terms in $T'^{(3)}$ is unifiable the algorithm terminates and returns

$$\begin{aligned} T_0 = T'^{(3)} = \{&a,\ g(f(x,a)),\ g(a),\ g(g(x)),\ g(f(x,g(y))),\ g(f(x,f(y,z)),\\ &f(a,g(x)),\ f(f(x,y),a),\ f(a,a),\ f(g(u),g(y)),\ f(f(x,y),g(z)),\\ &f(g(x),a),\ f(g(x),f(y,z))\}.\end{aligned}$$

It is not hard to verify that T_O is an orthogonal extension of T.

LEMMA 4.14
For each finite set T of linear terms the function *Orthogonalize* computes a finite, orthogonal extension T_O of T.

PROOF. It follows immediately from the definition of $co_H(t)$ and T that T' is finite and contains only linear terms at any stage of the algorithm. Therefore, if *Orthogonalize* terminates then T_O is finite and linear, too.

By Lemma 4.12, if t is an instance of s then $G_H(\{s\}) = G_H(\{t\}) \cup G(co_H(t|s))$. In particular we have

$$G_H(\{t_1,t_2\}) = G_H(\{t_1\theta(=t_2\theta)\}) \cup G_H(co_H(t_1\theta|t_1)) \cup G_H(co_H(t_1\theta|t_2)),$$

where θ is an mgu of t_1 and t_2.

It follows that $G_H(T') = G_H(T)$ is a loop–invariant of *Orthogonalize*. Therefore T_O represents $G_H(T)$. Since, by the **while**-condition, the terms in T_O are pairwise not unifiable the representation is orthogonal. Since for any t_i that is removed from T' the instance $t_i\theta$ is added to T' we conclude that T_O is an extension of T. We have thus proved the correctness of the algorithm.

To prove the termination of *Orthogonalize* we first show that there is only a fixed, finite number of terms that may occur in T_O. In fact the maximal depth of occurrence of a non-variable term (in terms in T') cannot increase. More formally, let

$$\tau_c(t) = \max\{|p| \mid p \in \mathcal{P}(t) \text{ but } [t|p] \notin V(t)\}.$$

If T is a set of terms then define

$$\tau_c(T) = \max\{\tau_c(t) \mid t \in T\}.$$

By definition of $co_H(t)$ we have $\tau_c(co_H(t)) = \tau_c(t)$. Moreover, for all linear terms t_1, t_2 that are unifiable by an mgu θ we have $\tau_c(t_1\theta(= t_2\theta)) = \tau_c(\{t_1, t_2\})$ (see [7] for details). Therefore $\tau_c(T')$ does not increase throughout the execution of the algorithm. But since we identify terms that are equal up to renaming of variables, and since we assume that H is based on a finite alphabet this means that there are only finitely many different possible values for T'.

We still have to exclude the possibility that *Orthogonalize* enters an infinite loop (i.e. that T' takes the same value infinitely many times): That this cannot happen is easily seen by observing the fact that in each traversal of the **while**-loop at least one term in T' is replaced by *proper* instances of it: if t_1 and t_2 are unifiable by the mgu θ then $t_1\theta = t_2\theta$ is a proper instance either of t_1 or of t_2, unless t_1 and t_2 are equal (up to a renaming of variables). But any two terms in the set T' are assumed to be different. The elements of $co_H(t_1\theta|t_1)$ and $co_H(t_1\theta|t_2)$ consist of proper instances of t_1 and t_2, respectively, by definition. It follows that, throughout the whole execution of the algorithm, T' never takes the same value twice. ∎

As a remark to the complexity of the algorithm, observe that $|co_H(t)|$ may be exponential (in the length of t.)

EXAMPLE 4.15
Let $t = f(a, \ldots, a)$ for some n-ary function symbol f. Then $co(t) = \{a\} \cup \{f(t_1, \ldots, t_n) \mid t_i$ of form $f(y_1, \ldots, y_n)$ or a, but at least one $t_i \neq a\}$ and thus $|co(t)| = 2^n$.

The following example shows that not only our algorithm, but the problem of orthogonalization itself is of exponential space (and time) complexity. Still, experiments with implementations of *Orthogonalize* show that for many relevant cases orthogonal extensions can be computed very rapidly.

EXAMPLE 4.16
Let T be the set of terms $\{f(a, x_2, x_3, \ldots, x_n),\ f(x_1, a, x_3, \ldots, x_n),\ \ldots,\ f(x_1, x_2, \ldots, x_{n-1}, a)\}$. Let H be based on the n-ary function symbol f and the constant symbol a. Clearly, T represents the set of all f-terms where at least one argument equals a. Of course T is not orthogonal since all its elements are pairwise unifiable. *Orthogonalize* constructs the set of terms $T_O = \{a\} \cup \{f(t_1, \ldots, t_n)\} \mid t_i$ of form $f(y_1, \ldots, y_n)$ or a but at least one $t_i = a\}$. Therefore $|T_O| = 2^n$ (in contrast to $|T| = n$). However, it is not hard to see that not only T_O (like any other orthogonal extension of T) but *any* set of terms that orthogonally represents $G_H(T)$ contains at least 2^n elements.

5 Constructing atomic representations

In the previous section we suggested representing Herbrand models by (finite) sets of atoms. This representation not only seems to be very natural, but is also close to termination sets of hyperresolution. Observe, that if R_{HS} terminates on a set $\mathcal{C}$ of Horn clauses then it produces an atomic representation of a Herbrand model of $\mathcal{C}$. This is trivially so, since positive hyperresolution produces only positive unit clauses (or $\Box$). Therefore the union over all hyperresolvents (i.e. the derivable positive clauses) constitutes an atomic representation of a (minimal) Herbrand model of $\mathcal{C}$. In this section we extend the method to non-Horn termination sets of hyperresolution.

DEFINITION 5.1
A set of clauses $\mathcal{C}$ is *(R_{HS}-)stable* if $R_{HS}(\mathcal{C}) = \mathcal{C}$.

Observe that the deductive closure $R^*_{HS}(\mathcal{C})$ of $\mathcal{C}$ is always stable. Our investigations mainly focus on finite stable sets, i.e. finite fixed points of R_{HS}. The following lemma is a standard result for Horn logic, but our formulation is slightly more general.

LEMMA 5.2
Let $\mathcal{C}$ be a finite set of non-positive clauses and let $\mathcal{A}$ be a finite set of atoms over the signature of $\mathcal{C}$ s.t. $\mathcal{C} \cup \mathcal{A}$ is stable and satisfiable. Then $\mathcal{A}$ is an ARM of $\mathcal{C} \cup \mathcal{A}$ (see Definition 4.1).

PROOF. Suppose, on the contrary, that $\mathcal{A}$ is not an ARM of $\mathcal{C} \cup \mathcal{A}$. Then the interpretation induced by $\mathcal{A}$ falsifies $\mathcal{C} \cup \mathcal{A}$. Thus the set $\mathcal{C} \cup \mathcal{A} \cup \{\neg P | P \in \mathbf{HB}(\mathcal{C}) - INT(A)\}$ is unsatisfiable.

By the compactness theorem of first-order logic there exists a finite $\mathcal{F} \subseteq \{\neg P | P \in \mathbf{HB}(\mathcal{C}) - INT(A)\}$ s.t. $\mathcal{C} \cup \mathcal{A} \cup \mathcal{F}$ is unsatisfiable. $\mathcal{F}$ cannot be empty as, by assumption, $\mathcal{C} \cup \mathcal{A}$ is satisfiable.

Because R_{HS} is complete we conclude $\Box \in R^*_{HS}(\mathcal{C} \cup \mathcal{A} \cup \mathcal{F})$. Because $\Box \notin \mathcal{C} \cup \mathcal{A} \cup \mathcal{F}$ there exists an $i \geq 0$ s.t. $\Box \in R^{i+1}_{HS}(\mathcal{C} \cup \mathcal{A} \cup \mathcal{F}) - R^i_{HS}(\mathcal{C} \cup \mathcal{A} \cup \mathcal{F})$. Let us abbreviate $R^i_{HS}(\mathcal{C} \cup \mathcal{A} \cup \mathcal{F})$ by $\mathcal{R}^i$. By definition of R^n_{HS} we obtain $\Box \in sub(\mathcal{R}^i \cup \rho_H(\mathcal{R}^i))$ and, in particular, $\Box \in \mathcal{R}^i \cup \rho_H(\mathcal{R}^i)$. Since $\Box \notin \mathcal{R}^i$ this implies $\Box \in \rho_H(\mathcal{R}^i)$.

By definition of ρ_H, and because $\mathcal{F}$ consists of ground literals only, there must be a $Q \in \mathcal{A}$ and a ground substitution γ over $H(\mathcal{C})$ s.t. $\neg Q\gamma \in \mathcal{F}$. Note that $\mathcal{C} \cup \mathcal{A}$ is stable and thus $\Box \notin R_{HS}(\mathcal{C} \cup \mathcal{A})$. But $Q\gamma \in INT(A)$ and therefore $\neg Q\gamma \notin \mathcal{F}$. By this contradiction we may conclude that $\mathcal{C} \cup \mathcal{A} \cup \{\neg P | P \in \mathbf{HB}(\mathcal{C}) - INT(A)\}$ is satisfiable, i.e. $\mathcal{A}$ is an ARM of $\mathcal{C} \cup \mathcal{A}$. ■

Lemma 5.2 suggests the following strategy for finding a model of $\mathcal{C}$: suppose $R^*_{HS}(\mathcal{C})$ is finite. Find a finite set of atoms $\mathcal{A}$ s.t. $NP(\mathcal{C}) \cup \mathcal{A}$ is stable, satisfiable and implies $\mathcal{C}$. Then $\mathcal{A}$ is an ARM of $NP(\mathcal{C}) \cup \mathcal{A}$ and also of $\mathcal{C}$ itself.

DEFINITION 5.3
We call $\mathcal{C}$ positively decomposed (and write $\mathcal{C} \in PDC$) iff $R^*_H(\mathcal{C})$ is finite and all clauses in $P(R^*_H(\mathcal{C}))$ are decomposed.

PDC contains all sets of Horn clauses $\mathcal{C}$ where R^*_H is finite. But also PVD$_+$ and OCC1N$_+$ are subsets of PDC and all classes that are shown to be decidable by hyperresolution in [7] and [12].

The following lemma provides the key technique for the reduction of a set of positive clauses to an atomic representation. We show that, for stable sets of clauses, positive clauses can be replaced by proper subclauses under preservation of satisfiability.

LEMMA 5.4
Let $\mathcal{C} \in PDC$ s.t. $\mathcal{C}$ is stable and contains a positive non-unit clause D. For any atom P in D we have

(1) $(\mathcal{C} - \{D\}) \cup \{P\}$ is satisfiable iff $\mathcal{C}$ is satisfiable, and
(2) $(\mathcal{C} - \{D\}) \cup \{P\}$ implies $\mathcal{C}$.

REMARK 5.5
(1) and (2) together guarantee that, for satisfiable $\mathcal{C}$, there exists a model of $(\mathcal{C} - \{D\}) \cup \{P\}$ which is also a model of $\mathcal{C}$.

PROOF of Lemma 5.4. (2) trivially holds since P implies D.

It remains to prove (1). By (2) it suffices to show that the satisfiability of $\mathcal{C}$ implies that of $\mathcal{C}'$ where $\mathcal{C}' = (\mathcal{C} - \{D\}) \cup \{P\}$. We proceed indirectly and assume that $\mathcal{C}$ is satisfiable but $\mathcal{C}'$ is unsatifiable. Then $\Box \in R_H^*(\mathcal{C}')$. (Note that we use the monotonic hyperresolution operator R_H and not R_{HS}.)

Let $D' = D - P$. We introduce a relation '$\leq_{D'}$' on all pairs of clauses (F, G): $F \leq_{D'} G$ iff either $F =_c G$ or there exists a renaming substitution η s.t. $V(D'\eta) \cap V(F) = \emptyset$ and $F \vee D'\eta =_c G$. We extend '$\leq_{D'}$' to sets of clauses $\mathcal{C}$, $\mathcal{D}$ by defining $\mathcal{C} \leq_{D'} \mathcal{D}$ iff for all $F \in \mathcal{C}$ there exists a $G \in \mathcal{D}$ s.t. $F \leq_{D'} G$.

We show (by induction on i) that

(1) $R_H^i(\mathcal{C}') \leq_{D'} R_H^*(\mathcal{C})$

for all $i \geq 0$. (Note that, since $\mathcal{C} \in PDC$, $R_H^*(\mathcal{C})$ is finite.)

$P \leq_{D'} D$ and $\mathcal{C}' - \{P\} = \mathcal{C} - \{D\}$, therefore $\mathcal{C}' \leq_{D'} \mathcal{C}$. But $\mathcal{C} \subseteq R_H^*(\mathcal{C})$ and thus, by definition of '$\leq_{D'}$', $\mathcal{C}' \leq_{D'} R_H^*(\mathcal{C})$. This proves (1) for $i = 0$.

As induction hypothesis assume that (1) holds for some i. Let $F \in R_H^{i+1}(\mathcal{C}')$. We only need to investigate the case where $F \in R_H^{i+1}(\mathcal{C}') - R_H^i(\mathcal{C}')$.

By definition of R_H there is a clash sequence $\Gamma = (C; D_1, \dots, D_n)$ over $R_H^i(\mathcal{C}')$ s.t. F is the clash resolvent defined by Γ. By the induction hypothesis there are clauses $H_1, \dots, H_n \in R_H^*(\mathcal{C})$ s.t.

$$D_j \leq_{D'} H_j \text{ for } 1 \leq j \leq n.$$

By definition of '$\leq_{D'}$' either $D_j =_c H_j$ or there exists a renaming substitution η_j s.t. $V(D'\eta_j) \cap V(D_j) = \emptyset$ and $H_j =_c D_j \vee D'\eta_j$. Note that $\Gamma' = (C; H_1, \dots, H_n)$ is a clash sequence over $R_H^*(\mathcal{C})$ (every nucleus clause C in $\mathcal{C}'$ is also contained in $R_H^*(\mathcal{C})$). By definition of the H_j Γ' defines a clash resolvent F' s.t. $F' =_c F \vee D'\eta_{i_1} \vee \dots \vee D'\eta_{i_k}$ for some $i_1, \dots, i_k \in \{1, \dots, n\}$. (Observe that if $D_1', \dots, D_n'$ are the factors of $D_1, \dots, D_n$ required to resolve the F from Γ then H_j' s.t. $H_j' =_c D_j' \vee D'\eta_j$ are the factors required to resolve F' from Γ'.) By definition of condensing we obtain $F' =_c F \vee D'\eta$ for some renaming substitution η s.t. $V(D'\eta_j) \cap V(F) = \emptyset$. Therefore we have $F' \leq_{D'} F$. This concludes the proof of the induction step. It follows $R_H^i(\mathcal{C}') \leq_{D'} R_H^*(\mathcal{C})$ for all i, i.e. $R_H^*(\mathcal{C}') \leq_{D'} R_H^*(\mathcal{C})$.

From the unsatisfiabiltiy of $\mathcal{C}'$ we infer $\Box \in R_H^*(\mathcal{C}')$. By definition of '$\leq_{D'}$' there must be a clause $H \in R_H^*(\mathcal{C})$ s.t. $\Box \leq_{D'} H$. This is only possible if $H =_c D'\eta$ for some renaming substitution η.

Since $R_{HS}^*(\mathcal{C}) \leq_s R_H^*(\mathcal{C})$ and since $\mathcal{C}$ is stable (i.e. $R_{HS}^*(\mathcal{C}) = \mathcal{C}$) we infer the existence of a clause $G \in \mathcal{C}$ s.t. $G \leq_{sub} H$. But $D =_c P \vee D'$ and thus $H \leq_{sub} D$ and, obviously, $H \neq_c D$.

It remains to show that D does not subsume $D'\eta$ (for any renaming substitution η): assume to the contrary that $LIT(P \vee D')\theta \subseteq LIT(D'\eta)$. Then also $LIT(P \vee D')\theta' \subseteq LIT(D')$ for some appropriate substitution θ'. This means that $P \vee D'$ is not condensed which contradicts the assumption that all clauses in $\mathcal{C}$, and in particular D, is condensed. It follows that $D \not\leq_{sub}$

$D'\eta$. But $G \leq_{sub} H$ and $H \leq_{sub} D$ implies $G \leq_{sub} D$. Moreover, $D \not\leq_{sub} G$ holds (note that $H =_c D'\eta$).

Summarizing we obtain two clauses $D, G \in \mathcal{C}$ s.t. $G \leq_{sub} D$ and $D \not\leq_{sub} G$. This condradicts the assumption that $\mathcal{C}$ is reduced with respect to subsumption ($\mathcal{C} = sub(\mathcal{C})$). So we conclude that the clause set $\mathcal{C}'$ is satisfiable. ■

REMARK 5.6
Lemma 5.4 relies essentially on the stability of the clause set $\mathcal{C}$. It is clearly wrong for non-stable clause sets: e.g. take $\mathcal{C} = \{\neg A, A \vee B\}$. $\mathcal{C}$ is satisfiable; but if we replace $A \vee B$ by A we obtain $\mathcal{C}' = \{\neg A, A\}$, which is unsatisfiable. Clearly, $\mathcal{C}$ is not stable; but its deductive closure $R^*_{HS}(\mathcal{C}) = \{\neg A, B\}$ obviously represents a Herbrand model of $\mathcal{C}$.

The replacement of $\mathcal{C}$ by $(\mathcal{C} - \{D\}) \cup \{P\}$ can be described by an operator α which selects a clause D in $\mathcal{C}$ and a literal P in D. The model construction consists in alternately applying transformation α and computing the deductive closure (w.r.t. R^*_{HS}) of the new set of clauses. (Note that $(\mathcal{C} - \{D\}) \cup \{P\}$ need not be stable, so we have to apply R_{HS} after selecting a literal.) We actually need an operator T transforming a stable set into a new stable set in which some non-unit positive clauses are replaced by unit clauses. We define

$$T(\mathcal{C}) = R^*_{HS}(\alpha(\mathcal{C})) \text{ for a stable set } \mathcal{C} \in PDC, \text{ and}$$

$$T^0(\mathcal{C}) = \mathcal{C}, \quad T^{i+1}(\mathcal{C}) = T(T^i(\mathcal{C})) \text{ for } i \geq 0.$$

Our aim is to show that for all $\mathcal{C} \in PDC$ there is a finite fixed point of T: i.e. there exists an i s.t. $T^{i+1}(\mathcal{C}) = T^i(\mathcal{C})$. Then the positive unit clauses of $T^i(\mathcal{C})$ represent a Herbrand model of $\mathcal{C}$.

EXAMPLE 5.7
Let

$$\begin{aligned} \mathcal{C} = \{&P(x) \vee Q(x) \vee \neg R(x) \vee \neg S(x), \\ &S(a) \vee E(a), \\ &R(a) \vee Q(a), \\ &\neg P(a) \vee \neg Q(a)\}. \end{aligned}$$

$\mathcal{C}$ is a non-stable set of clauses in PDC. We first apply R^*_{HS} to obtain a stable set.

$$\begin{aligned} R^*_{HS}(\mathcal{C}) = NP(\mathcal{C}) \cup \{&R(a) \vee Q(a), \\ &S(a) \vee E(a), \\ &P(a) \vee Q(a) \vee E(a)\}. \end{aligned}$$

We abbreviate $R^*_{HS}(\mathcal{C})$ by $\mathcal{C}_1$. Let $\alpha(\mathcal{C}_1) = (\mathcal{C}_1 - \{S(a) \vee E(a)\}) \cup S(a)$. By Lemma 5.4 we know that $\alpha(\mathcal{C}_1)$ is satisfiable and all its models are models of $\mathcal{C}$, too.

$\alpha(\mathcal{C}_1)$ is not stable, therefore we have to compute its closure w.r.t. R^*_{HS}. Let $\mathcal{C}_2 = T(\mathcal{C}_1)$ $(= R^*_{HS}(\alpha(\mathcal{C}_1)))$. Then

$$\begin{aligned} \mathcal{C}_2 = NP(\mathcal{C}) \cup \{&S(a), \\ &R(a) \vee Q(a), \\ &P(a) \vee Q(a)\}. \end{aligned}$$

If we now set $\alpha(\mathcal{C}_2) = (\mathcal{C}_1 - \{P(a) \vee Q(a)\}) \cup \{Q(a)\}$ we obtain

$$\begin{aligned} \mathcal{C}_3 = T(\mathcal{C}_2) = NP(\mathcal{C}) \cup \{&S(a), \\ &Q(a)\}. \end{aligned}$$

C_3 is irreducible w.r.t. α, i.e. $\alpha(C_3) = C_3$. Therefore also $T(C_3) = C_3$ and $\{S(a), Q(a)\}$ is an ARM of C_3. Since C_3 implies C_2, C_2 implies C_1, and C_1 implies C this is also an ARM of C.

In order to prove that the $T^i(\mathcal{C})$ converge to a stable clause set $\mathcal{D}$ s.t. $P(\mathcal{D})$ consists of unit clauses only we shall first show that $T(\mathcal{C})$ is strictly smaller than $R_H^*(\mathcal{C})$ w.r.t. some Noetherian ordering $\prec$.

DEFINITION 5.8
Let $\mathcal{C}, \mathcal{D}$ be two clause sets (i.e. finite sets of condensed clauses). $\mathcal{C} \prec \mathcal{D}$ iff

(1) $\mathcal{C} \leq_{sub} \mathcal{D}$,
(2) for all $C \in \mathcal{C}$ there exists a $D \in \mathcal{D}$ s.t. $C \leq_{sub} D$ and $|C| \leq |D|$, and
(3) $\mathcal{D} \not\leq_{sub} \mathcal{C}$.

PROPOSITION 5.9
The relation $\prec$ is irreflexive and transitive.

PROOF. Irreflexivity is trivial since $\mathcal{C} \leq_{sub} \mathcal{C}$ for every clause set $\mathcal{C}$. It remains to show transitivity.

Let $\mathcal{C}_1, \mathcal{C}_2$ and $\mathcal{C}_3$ be sets of condensed clauses s.t. $\mathcal{C}_1 \prec \mathcal{C}_2 \prec \mathcal{C}_3$. Then, by the transitivity of $\leq_{sub}$ and part (1) of Definition 5.8, we have $\mathcal{C}_1 \leq_{sub} \mathcal{C}_3$. To establish condition (2) consider any $C_1 \in \mathcal{C}_1$. By part (2) of Definition 5.8 there exists a clause $C_2 \in \mathcal{C}_2$ s.t. $C_1 \leq_{sub} C_2$ and $|C_1| \leq |C_2|$. Similarily, for C_2 we obtain a clause $C_3 \in \mathcal{C}_3$ s.t. $C_2 \leq_{sub} C_3$ and $|C_2| \leq |C_3|$. Both, $\leq_{sub}$ and $\leq$, are transitive and thus (2) also holds for $\mathcal{C}_1$ and $\mathcal{C}_3$.

We still have to show $\mathcal{C}_3 \not\leq_{sub} \mathcal{C}_1$. Let as assume $\mathcal{C}_3 \leq_{sub} \mathcal{C}_1$ then, by $\mathcal{C}_1 \leq_{sub} \mathcal{C}_2$ and the transitivity of $\leq_{sub}$ we get $\mathcal{C}_3 \leq_{sub} \mathcal{C}_2$. But this contradicts the assumption that $\mathcal{C}_2 \prec \mathcal{C}_3$. (See part (3) of Definition 5.8.) Thus $\mathcal{C}_3 \not\leq_{sub} \mathcal{C}_1$ and, putting things together $\mathcal{C}_1 \prec \mathcal{C}_3$. ∎

LEMMA 5.10
The relation $\prec$ is Noetherian: i.e. there exists no infinite strictly descending chain $\mathcal{C}_0 \succ \mathcal{C}_1 \succ \ldots \mathcal{C}_i \succ \mathcal{C}_{i+1} \succ \ldots$ (for condensed sets of clauses $\mathcal{C}_i$).

PROOF. Consider a descending chain $\mathcal{C}_0 \succ \mathcal{C}_1 \succ \ldots$. We assume that the chain is infinite and derive a contradiction.

Using a simple induction argument it follows from Definition 5.8 that, for all $i \geq 0$, the size of the clauses in $\mathcal{C}_i$ cannot exceed the maximal size of the clauses in $\mathcal{C}_0$. Again by the definition of $\prec$, every $\mathcal{C}_i$ must contain a clause C s.t. $C \leq_{sub} D$ for some $D \in \mathcal{C}_{i-1}$ and $|C| \leq |D|$ but $\mathcal{C}_{i-1} \not\leq_{sub} \{C\}$. Since every clause subsumes itself, C is not contained in $\mathcal{C}_{i-1}$. By part (1) of Definition 5.8 we obtain $\mathcal{C}_{i-1} \leq_{sub} \mathcal{C}_k$ and thus $\mathcal{C}_k \not\leq_{sub} \{C\}$ for all $k \leq i-1$. Therefore, we obtain a clause C_i for every $i \geq 0$ s.t. $\bigcup_{0 \leq j \leq i-1} \mathcal{C}_j \not\leq_{sub} \{C_i\}$. Let us consider the infinite sequence $C_0, C_1, \ldots, C_i, \ldots$ of these clauses. Let $\mathcal{C}_0 = \{D_1, \ldots, D_m\}$. Then for every C_i there is some $D_j \in \mathcal{C}_0$ s.t. $C_i \leq_{sub} D_j$, (see part (2) of Definition 5.8). By the (transfinite) pigeon hole principle there exists a clause $D_r \in \mathcal{C}_0$ s.t. $C_i \leq_{sub} D_r$ for infinitely many clauses C_i, which are (by construction) pairwise different. Moreover, $|C_i| \leq d$ for all i and some constant d (take $d = \max_{C \in \mathcal{C}_0}(|C|)$). But there are only finitely many different condensed clauses with a fixed bound on their size and term depth. Thus we obtain a contradiction and may conclude that $\prec$ is Noetherian. ∎

LEMMA 5.11
Let $\mathcal{C}$ be a stable, satisfiable set in PDC containing a positive nonunit clause. Then $R_H^*(T(\mathcal{C})) \prec R_H^*(\mathcal{C})$.

PROOF. We first show that $R^*_H(\alpha(\mathcal{C})) \prec R^*_H(\mathcal{C})$. (Note that $T(\mathcal{C}) = R^*_{HS}(\alpha(\mathcal{C}))$.)

By definition of α we have $\alpha(\mathcal{C}) \leq_{sub} \mathcal{C}$. Because R_H preserves the subsumption relation for all clause sets we obtain $R^*_H(\alpha(\mathcal{C})) \leq_{sub} R^*_H(\mathcal{C})$. (This is condition (1) of Definition 5.8.)

To prove condition (3) of Definition 5.8 we show that there exists a $C \in R^*_H(\alpha(\mathcal{C}))$ s.t. $R^*_H(\mathcal{C}) \not\leq_{sub} \{C\}$. Let E be the positive non-unit clause of $\mathcal{C}$ selected by α. Then there is an atom P in E s.t. $\alpha(\mathcal{C}) = (\mathcal{C} - \{E\}) \cup \{P\}$. Since $\mathcal{C}$ is satisfiable, $\alpha(\mathcal{C})$ is satisfiable too, by Lemma 5.4. Therefore $\Box \notin R^*_H(\alpha(\mathcal{C}))$.

We show (indirectly) that $R^*_H(\mathcal{C})$ does not subsume P. Assume that $R^*_H(\mathcal{C}) \leq_{sub} \{P\}$. By assumption $\mathcal{C}$ is stable, i.e. $R^*_{HS}(\mathcal{C}) = \mathcal{C}$. We conclude that $\mathcal{C} \leq_{sub} R^*_H(\mathcal{C})$ and also $\mathcal{C} \leq_{sub} \{P\}$. Thus there exists a clause $D = P_1 \vee \ldots \vee P_n$ in $\mathcal{C}$ s.t. $D \leq_{sub} P$. Note that D must contain a factor P_0 s.t. $P_0 \leq_{sub} P$. By definition, the clause E selected by α is condensed, contains P, but is not identical to P. Clearly $P \leq_{sub} E$ and thus $D \leq_{sub} E$. We show that D *properly* subsumes E. Suppose we have $E \leq_{sub} D$ then also $E \leq_{sub} P_0$ and consequently $E \leq_{sub} P$. But then E must contain a factor P'_0 s.t. $P'_0 \leq_{sub} P$. Since E contains P, P'_0 cannot be strictly more general than P and therefore $P'_0 =_{sub} P$. But then E is not condensed and we obtain a contradiction. We conclude that $E \not\leq_{sub} D$ and thus D and E are two different clauses in $\mathcal{C}$ s.t. $D \leq_{sub} E$. This, however, contradicts the stability of $\mathcal{C}$. Therefore $R^*_H(\mathcal{C}) \not\leq_{sub} \{P\}$ and we have established condition (3) of Definition 5.8.

It remains to show that condition (2) of Definition 5.8 also holds. We prove the following somewhat stronger proposition by induction:

(*) For all $n \geq 0$ and for all $C \in R^n_H(\alpha(\mathcal{C}))$ there exists a $D \in R^*_H(\mathcal{C})$ s.t. $|C| \leq |D|$ and a renaming substitution ν s.t. $LIT(C\nu) \subseteq LIT(D)$. (Moreover, if C is positive then D is positive, too.)

Clearly, the conditions on C and D in (*) imply $C \leq_{sub} D$.

If $n = 0$ choose $C = P$ and $D = E$. Then $LIT(P) \subseteq LIT(D)$ and $\alpha(\mathcal{C}) - \{C\} = \mathcal{C} - \{E\}$.

(IH) Suppose (*) holds for n.

Let $C \in R^{n+1}_H(\alpha(\mathcal{C}))$. If C is already in $R^n_H(\alpha(\mathcal{C}))$ we may apply (IH) and obtain (*) for $n+1$. Thus it suffices to consider the case where $C \in R^{n+1}_H(\alpha(\mathcal{C})) - R^n_H(\alpha(\mathcal{C}))$.

By definition of R_H, C must be a resolvent defined by a clash $\Gamma = (C_0; D_1, \ldots, D_n)$ s.t. $C_0 \in NP(\mathcal{C})$ and $D_i \in R^n_H(\alpha(\mathcal{C}))$ for $1 \leq i \leq n$. By (IH) there are positive clauses $E_1, \ldots, E_n \in R^*_H(\mathcal{C})$ s.t. there exist renaming substitutions ν_i with $LIT(D_i\nu) \subseteq LIT(E_i)$ and $|D_i| \leq |E_i|$. Let Γ' be the clash $(C_0; E_1, \ldots, E_n)$ over $R^*_H(\mathcal{C})$. Because every E_i contains a variant D'_i of D_i we can simulate factoring and the selection of atoms for binary resolution from D_i in D'_i and thus within E_i. Therefore there exists a clash resovent E defined by Γ' and a renaming substitution ν s.t. $LIT(C\nu) \subseteq LIT(E)$. Clearly, we also have $|C| \leq |E|$. By $\rho_H(R^*_H(\mathcal{C})) \subseteq R^*_H(\mathcal{C})$ we obtain $E \in R^*_H(\mathcal{C})$. Thus, by induction, we obtain for all $C \in R^*_H(\alpha(\mathcal{C}))$ a clause $D \in R^*_H(\mathcal{C})$ s.t. $C \leq_{sub} D$. This gives condition (2) of the definition of $\prec$ and, eventually, $R^*_H(\alpha(\mathcal{C})) \prec R^*_H(\mathcal{C})$.

It remains to show that $R^*_H(T(\mathcal{C})) \prec R^*_H(\mathcal{C})$: by definition of T we have $T(\mathcal{C}) = R^*_{HS}(\alpha(\mathcal{C}))$. But this implies

$$R^*_H(T(\mathcal{C})) = R^*_H(R^*_{HS}(\alpha(\mathcal{C}))) \subseteq R^*_H(R^*_H(\alpha(\mathcal{C}))) = R^*_H(\alpha(\mathcal{C})).$$

Moreover, by definition of R_H and R_{HS}, we have

$$R^*_{HS}(\alpha(\mathcal{C})) =_{sub} R^*_H(\alpha(\mathcal{C}))$$

and thus also $R^*_H(T(\mathcal{C})) =_{sub} R^*_H(\alpha(\mathcal{C}))$. But $R^*_H(T(\mathcal{C})) \subseteq R^*_H(\alpha(\mathcal{C}))$, $R^*_H(T(\mathcal{C})) =_{sub} R^*_H(\alpha(\mathcal{C}))$ and $R^*_H(\alpha(\mathcal{C})) \prec R^*_H(\mathcal{C})$ together immediately imply

$$R^*_H(T(\mathcal{C})) \prec R^*_H(\mathcal{C}).$$

∎

Looking at Lemma 5.11 one may ask, why we have proven only $T(\mathcal{C})) \prec R^*_H(\mathcal{C})$ instead of $T(\mathcal{C}) \prec \mathcal{C}$. The simple answer is that the last relation does not always hold.

EXAMPLE 5.12
Let

$$\mathcal{C} = \{R(z) \vee S(v),\ Q(x) \vee \neg P(x) \vee \neg T(x),\ T(y) \vee R(z),\ P(u) \vee S(v)\}.$$

$\mathcal{C}$ is in PDC and is stable. (Note that the only hyperresolvent $Q(x) \vee R(z) \vee S(v)$ is subsumed by $R(z) \vee S(v)$. Let

$$\alpha(\mathcal{C}) = \{R(z) \vee S(v),\ Q(x) \vee \neg P(x) \vee \neg T(x),\ T(y) \vee R(z),\ P(u)\}.$$

Then $R^*_{HS}(\alpha(\mathcal{C})) = \alpha(\mathcal{C}) \cup \{Q(x) \vee R(z)\}$.

Note that $Q(x) \vee R(z)$ is not subsumed by any clause in $\alpha(\mathcal{C})$. Moreover, there is no clause $D \in \mathcal{C}$ s.t. $Q(x) \vee R(z) \leq_{sub} D$. By definition of $\prec$, we conclude $T(\mathcal{C}) \not\prec \mathcal{C}$.

On the other hand, $T(\mathcal{C})) \prec R^*_H(\mathcal{C})$ since the resolvent $Q(x) \vee R(z) \vee S(v)$ is not deleted by R_H and thus can serve as a 'bound' for $Q(x) \vee R(z)$.

The algorithm for constructing an ARM essentially consists in computing the sequence $T^1(\mathcal{C}), T^2(\mathcal{C}), \dots$ for some R_{HS}-stable set of clauses $\mathcal{C}$. From Lemma 5.4 we know that T is correct. It remains to prove termination, i.e. to show the existence of a finite fixed point $T^i(\mathcal{C}) = T^{i+1}(\mathcal{C})$. Unfortunately, Lemma 5.11 does not directly yield $T(\mathcal{C}) \prec \mathcal{C}$ but merely guarantees $T(\mathcal{C}) \prec R^*_H(\mathcal{C})$. Nevertheless the sequence $T^1(\mathcal{C}), T^2(\mathcal{C}), \dots$ 'converges' to some $T^i(\mathcal{C})$ for stable clause sets $\mathcal{C} \in PDC$.

THEOREM 5.13
Let $\mathcal{C}$ be a stable, satisfiable clause set in PDC. Then there exists an i s.t. $T^i(\mathcal{C}) = T^{i+1}(\mathcal{C})$. Moreover, $P(T^i(\mathcal{C}))$ consists of unit clauses only and is an atomic representation of a Herbrand model of $\mathcal{C}$.

PROOF. Let us assume that there exists an i s.t. $T^i(\mathcal{C}) = T^{i+1}(\mathcal{C})$. Then, by Lemma 5.4, $T^i(\mathcal{C})$ implies $\mathcal{C}$ and $T^i(\mathcal{C})$ is satisfiable. Moreover, $P(T^i(\mathcal{C}))$ consists of unit clauses only, since otherwise $R^*_H(\alpha(T^i(\mathcal{C}))) \prec R^*_H(T^i(\mathcal{C}))$ and by Defnition 5.8, part (3) $R^*_H(T^i(\mathcal{C})) \not\leq_{sub} R^*_H(\alpha(T^i(\mathcal{C})))$. But $T^i(\mathcal{C}) \leq_{sub} R^*_H(T^i(\mathcal{C}))$ and $R^*_{HS}(\alpha(T^i(\mathcal{C}))) \subseteq R^*_H(\alpha(T^i(\mathcal{C})))$, which yields $R^*_H(\alpha(T^i(\mathcal{C}))) =_{sub} T^{i+1}(\mathcal{C})$. Therefore $P(T^i(\mathcal{C}))$ is an atomic representation of a Herbrand model of $\mathcal{C}$.

It remains to show that there indeed exists a number i s.t. $T^i(\mathcal{C}) = T^{i+1}(\mathcal{C})$. By Lemma 5.11 we know that $R^*_H(T(\mathcal{C})) \prec R^*_H(\mathcal{C})$ if $\mathcal{C}$ is stable, satisfiable and contains positive non-unit clauses. By Lemma 5.4 we obtain that $\alpha(\mathcal{C})$ is satisfiable and thus $T(\mathcal{C})$ is satisfiable and R_{HS}-stable. Thus, by iterating this argument, we obtain a descending chain

$$\dots \prec R^*_H(T^{i+1}(\mathcal{C})) \prec R^*_H(T^i(\mathcal{C})) \prec \dots \prec R^*_H(\mathcal{C}).$$

By Lemma 5.10 the relation $\prec$ is Noetherian, i.e. there must exist a minimal element, say $R^*_H(T^k(\mathcal{C}))$, in the above chain. We claim that $T^{k+1}(\mathcal{C}) = T^k(\mathcal{C})$; because, if they were different, $\alpha(T^k(\mathcal{C}))$ would be different from $T^k(\mathcal{C})$. This, in turn, would imply the existence of a positive non-unit clause in $T^k(\mathcal{C})$ and, by Lemma 5.11, $R^*_H(T^{k+1}(\mathcal{C})) \prec R^*_H(T^k(\mathcal{C}))$, which contradicts the minimality of $R^*_H(T^k(\mathcal{C}))$. Therefore we obtain $T^{k+1}(\mathcal{C}) = T^k(\mathcal{C})$. ∎

6 Equivalence of atomic representations

In this section we develop an algorithm that decides whether two finite sets of atoms represent the same Herbrand model. Observe that a set of atoms represents a Herbrand model only with reference to a Herbrand universe. We did not have to mention the universe explictly in the investigations of the previous sections, since we implictly always referred to the Herbrand universe of the clause set for which we wanted to construct a model representation. In the following, we have to make the reference to Herbrand universes H explicit, but implicitly assume that H is based on a signature that contains at least the symbols occurring in the clauses in question.

EXAMPLE 6.1
$\mathcal{A}_1 = \{P(x,a)\}$ and $\mathcal{A}_2 = \{P(a,a), P(f(x),a)\}$ are equivalent w.r.t. the Herbrand universe $H = \{f^i(a) \mid i \geq 0\}$: the set of all H-instances of $P(x,a)$ is identical with the set of all H-instances of $P(f(x),a)$ augmented by $P(a,a)$. But $\mathcal{A}_1$ and $\mathcal{A}_2$ do not represent the same Herbrand model w.r.t. $H' = H \cup \{f^i(b) \mid i \geq 0\}$, since, for example, $P(b,a)$ is true in $INT_{H'}(\mathcal{A}_1)$ but false in $INT_{H'}(\mathcal{A}_2)$. Note that subsumption tests do not suffice to discover the equivalence of $\mathcal{A}_1$ and $\mathcal{A}_2$ w.r.t. H: we have $\mathcal{A}_1 \leq_{sub} \mathcal{A}_2$ but not $\mathcal{A}_2 \leq_{sub} \mathcal{A}_1$.

Clearly, for all finite sets of atoms, $\mathcal{A}_1 =_{sub} \mathcal{A}_2$ implies the equivalence of $\mathcal{A}_1$ and $\mathcal{A}_2$ w.r.t. every Herbrand universe. (In this case $\mathcal{A}_1$ and $\mathcal{A}_2$ are logically equivalent.) However, to decide the equivalence w.r.t. specific Herbrand universes we need a weaker, parameterized form of subsumption.

DEFINITION 6.2
Let H be a set of ground terms and C, D be clauses. We say that C H-subsumes D and write $C \leq_{sH} D$ if C subsumes all H-instances of D. We extend this relation to sets of clauses as follows: $\mathcal{C} \leq_{sH} \mathcal{D}$ if for all H-instances $D\gamma$ of all $D \in \mathcal{D}$ there exists a $C \in \mathcal{C}$ s.t. $C \leq_{sub} D\gamma$. We write $\mathcal{C} =_{sH} \mathcal{D}$ if $\mathcal{C} \leq_{sH} \mathcal{D}$ and $\mathcal{D} \leq_{sH} \mathcal{C}$.

Note that $\mathcal{C} \leq_{sH} \mathcal{D}$ does not imply that all clauses in $\mathcal{D}$ are H-subsumed by some clause in $\mathcal{C}$. The subsumption relation is stronger than H-subsumption, i.e. $\mathcal{C} \leq_{sub} \mathcal{D}$ implies $\mathcal{C} \leq_{sH} \mathcal{D}$ for all sets of ground terms H.

The following proposition follows immediately from the definition of H-subsumption and INT_H:

PROPOSITION 6.3
Let $\mathcal{A}$, $\mathcal{B}$ be a finite set of atoms and H be a Herbrand universe. Then $\mathcal{A}$ and $\mathcal{B}$ represent the same Herbrand model w.r.t. H iff $\mathcal{A} =_{sH} \mathcal{B}$.

EXAMPLE 6.4
Let $\mathcal{A}_1$, $\mathcal{A}_2$ and H be as above. Since $\mathcal{A}_1 \leq_{sub} \mathcal{A}_2$ we also have $\mathcal{A}_1 \leq_{sH} \mathcal{A}_2$. But although $\mathcal{A}_2 \not\leq_{sub} \mathcal{A}_1$ we can show that $\mathcal{A}_2 \leq_{sH} \mathcal{A}_1$.

We distinguish two types of H-substitutions θ:

(1) $\theta(x) = a$: then $P(x,a)\theta = P(a,a)$. Thus, since $P(a,a) \in \mathcal{A}_2$ and $P(a,a) \leq_{sub} P(a,a)$, we have $\mathcal{A}_2 \leq_{sH} P(x,a)\theta$.

(2) $\theta(x) = f(t)$ for some $t \in H$: $P(f(x),a) \in \mathcal{A}_2$ and $P(f(x),a) \leq_{sub} P(f(t),a)$ for all t therefore $\mathcal{A}_2 \leq_{sH} P(x,a)\theta$ also in this case.

We have thus demonstrated that $\mathcal{A}_2 \leq_{sH} \mathcal{A}_1$. But note that neither $\{P(a,a)\} \leq_{sH} \mathcal{A}_1$ nor $\{P(f(x),a)\} \leq_{sH} \mathcal{A}_1$. Moreover, if we again consider $H' = H \cup \{f^i(b) \mid i \geq 0\}$ we have $\mathcal{A}_2 \not\leq_{sH'} \mathcal{A}_1$.

Our aim is to provide an algorithm that decides whether two finite sets of atoms represent the same model. By Proposition 6.3 it is sufficient to show that H-subsumption is decidable. Lemma 6.5 and Theorem 6.6 show that we may reduce H-subsumption to a number of (ordinary) subsumption tests.

LEMMA 6.5
Let $\mathcal{C}, \mathcal{D}$ be clause sets and H be an infinite Herbrand universe. Suppose that for all $D \in \mathcal{D}$ and all $x \in V(D)$ $\tau_{\min}(x,D) > \tau(\mathcal{C})$. Then $\mathcal{C} \leq_{sH} \mathcal{D}$ implies $\mathcal{C} \leq_{sub} \mathcal{D}$.

PROOF. We have to show that for every $D \in \mathcal{D}$ there exists a $C \in \mathcal{C}$ s.t. $C \leq_{sub} D$.

Let D be a clause in $\mathcal{D}$ and $V(D) = \{x_1, \ldots, x_m\}$. If $V(D) = \emptyset$ then $D\theta = D$ for all θ and thus any clause subsuming $D\theta$ subsumes D, i.e. $\mathcal{C} \leq_{sub} \{D\}$. Therefore we may assume that D is non-ground (i.e. $m > 0$).

Since H is infinite (but based on a finite alphabet) it contains terms of arbitrary depths. Particularly, there are terms $t_1, \ldots, t_m \in H$ s.t. $\tau(t_1) > \tau(D)$ and $\tau(t_i) - \tau(t_{i-1}) > \tau(D)$ for all $1 < i \leq m$. Let $\theta = \{x_1 \leftarrow t_1, \ldots, x_m \leftarrow t_m\}$. By $\mathcal{C} \leq_{sH} \{D\}$ there exists a $C \in \mathcal{C}$ with $C \leq_{sub} D\theta$. We prove that C also subsumes D itself.

By $C \leq_{sub} D\theta$ there exist literals $L_1, \ldots, L_n$ in D and a substitution μ s.t. for all L in C we have $L\mu = L_j\theta$ for some $1 \leq j \leq n$. Let $\mathcal{P}_i = \bigcup_{L \in C_i} FRONT(L, L_i)$ and $\mathcal{P} = \bigcup_{1 \leq i \leq n} \mathcal{P}_i$: i.e. $\mathcal{P}$ is the set of pairs of terms on that L differs with any of the L_i. Let $\mathcal{P}'$ be defined like $\mathcal{P}$ where L_i is replaced by $L_i\theta$. Observe that μ is defined to be a unifier of all pairs of terms $(s,t) \in \mathcal{P}'$. We investigate the set $\mathcal{P}$.

First note that, by the unifiability of the pairs in $\mathcal{P}'$, for all $(s,t) \in \mathcal{P}$ either s or t is a variable. But, by the assumption that $\tau_{\min}(x,D) > \tau(\mathcal{C})$ for all $x \in V(D)$ we conclude that t cannot be a variable. Thus all pairs in $\mathcal{P}$ are of the form (y,s) where $y \in V(C)$ and s is a non-variable term occurring in D. (Moreover, $\mathcal{P}$ cannot be empty because of $\tau_{\min}(x,D) > \tau(\mathcal{C})$.)

For all $y \in V(C)$ let $\mathcal{P}(y) = \{s \mid (y,s) \in \mathcal{P}\}$ (i.e. the set of terms corresponding to y in some L_i). If for all $y \in V(C)$ the sets $\mathcal{P}(y)$ are singletons then $C \leq_{sub} D$. For if $\mathcal{P}(y) = \{s_y\}$ then the matching substitution $\lambda = \{y \leftarrow s_y \mid y \in V(C)\}$ fulfills $y\lambda = s$ for all pairs $(y,s) \in \mathcal{P}\}$. Consequently we obtain $LIT(C)\lambda \subseteq \{L_1, \ldots, L_n\}$ which means $C \leq_{sub} D$.

We show that indeed $|\mathcal{P}(y)| = 1$. Suppose, on the contrary, that (y,s_1) and (y,s_2) with $s_1 \neq s_2$ are in $\mathcal{P}$ for some $y \in V(C)$. By $C \leq_{sub} D\theta$ we have $s_1\theta = s_2\theta$, i.e. s_1 and s_2 are unifiable by θ. Since s_1 and s_2 are different but unifiable $FRONT(s_1,s_2)$ consists of pairs of the form (x_i,t) or (t,x_i). (Recall that $V(D) = \{x_1, \ldots, x_m\}$.) But by the definition of θ we have $\tau(x_i\theta) > \tau(t\theta)$ if $V(t) \subseteq \{x_1, \ldots, x_{i-1}\}$ and $\tau(x_i\theta) < \tau(t\theta)$ if $V(t) \cap \{x_i, \ldots, x_m\} \neq \emptyset$. Therefore θ is not a unifier of (x_i,t) (or (t,x_i) respectively.) But then s_1 and s_2 cannot be unifiable either. This contradiction implies that the $\mathcal{P}(y)$ are singletons and consequently $C \leq_{sub} D$ as shown above. ∎

THEOREM 6.6
Let $\mathcal{C}$, $\mathcal{D}$ be clause sets and H be a Herbrand universe. It is decidable whether $\mathcal{C} \leq_{sH} \mathcal{D}$

PROOF. If H is finite then the set $\mathcal{D}^*$ of all H-instances of clauses is finite, too. Thus, by definition of H-subsumption, we obtain $\mathcal{C} \leq_{sH} \mathcal{D}$ iff $\mathcal{C} \leq_{sub} \mathcal{D}^*$. But subsumption is well known to be decidable.

If H is infinite we apply 'partial saturation' to construct a finite set $\mathcal{D}^*$ s.t. $INT_H(\mathcal{D}) = INT_H(\mathcal{D}^*)$, but $\tau_{\min}(x, D) > \tau(\mathcal{C})$ for all non-ground clauses $D \in \mathcal{D}^*$. Clearly, $\mathcal{C} \leq_{sH} \mathcal{D}$ iff $\mathcal{C} \leq_{sH} \mathcal{D}^*$. Moreover, $\mathcal{C} \leq_{sH} \mathcal{D}^*$ iff $\mathcal{C} \leq_{sub} \mathcal{D}^*_g$ and $\mathcal{C} \leq_{sH} (\mathcal{D}^* - \mathcal{D}^*_g)$, where $\mathcal{D}^*_g$ is the set of ground clauses in $\mathcal{D}^*$. By Lemma 6.5 $\mathcal{C} \leq_{sH} (\mathcal{D}^* - \mathcal{D}^*_g)$ implies $\mathcal{C} \leq_{sub} (\mathcal{D}^* - \mathcal{D}^*_g)$. We conclude that $\mathcal{C} \leq_{sH} \mathcal{D}$ iff $\mathcal{C} \leq_{sub} \mathcal{D}^*$. This means that, also in this case, we have reduced H-subsumption to a finite number of subsumption tests. ∎

An ARM can also be considered as a finite set of unit clauses. Therefore Proposition 6.3 implies the following corollary:

THEOREM 6.7
Let $\mathcal{A}$, $\mathcal{B}$ be finite set of atoms and H be a Herbrand universe. It is decidable whether $\mathcal{A}$ and $\mathcal{B}$ represent the same Herbrand model w.r.t. H.

The algorithm for deciding the equivalence of atomic representations as suggested by the proof of Theorem 6.6 clearly is not very efficient. On the other hand the result is much more general than needed. Especially, if we concentrate on *linear* ARMs we might hope for more efficient algorithms. Below we show that the special properties of linear terms as discussed in Section 4 indeed allow the formulation of a decision algorithm for the equivalence of linear atomic representations that seems more suitable for implementations.

```
function Equivalent
{Input:  finite sets $T_1, T_2$ of linear expressions, alphabet of $H$}
{Output:  equivalence is true or false}
begin
   $T_1'$ := Orthogonalize($T_1$) (w.r.t. $H$);
   $T_2'$ := Orthogonalize($T_2$) (w.r.t. $H$);
   while $T_1' \neq T_2'$ do
      begin
      if $\exists t_1 \in T_1'$, $t_2 \in T_2'$ s.t. $t_1$ and $t_2$ are unifiable then
         $\theta$ := mgu of $t_1$ and $t_2$;
         $T_1' := (T_1' - \{t_1\}) \cup co_H(t_1\theta|t_1)$
         $T_2' := (T_2' - \{t_2\}) \cup co_H(t_1\theta|t_2)$
      else return false
      endif
      end { while };
   return true
end.
```

REMARK 6.8
For testing the equivalence of T_1 and T_2 as above we actually do not need orthogonal *extensions*; any orthogonalization of T_1 and T_2 would suffice. Some other improvements are possible as well. However, remember that even the smallest orthogonalization may contain exponentially many terms (compared to the number of terms in the original set). Thus the

run time of *Equivalent* is exponential in the worst case, even if these improvements are taken into account. G. Gottlob has demonstrated [8] that the problem of testing the equivalence of atomic representations is NP-hard even for linear representations; therefore we cannot expect to find a polynomial algorithm, anyway.

LEMMA 6.9
For any two finite sets T_1, T_2 of linear expressions the function *Equivalent* decides whether $G_H(T_1) = G_H(T_2)$.

PROOF. For the correctness of the algorithm we rely on the correctness of *Orthogonalize* and the properties of $co_H(t\theta|t)$ proved in Lemma 4.12. It follows that T_1' and T_2' are orthogonal throughout the execution of the program. Moreover, if T is an orthogonal representation of $G_H(T)$ then, for any $t \in T$ and any H-instance $t\theta$ of t, $(T - \{t\}) \cup co_H(t\theta|t)$ orthogonally represents $G_H(T) - G_H(t\theta)$. Therefore T_1' and T_2' are equivalent iff $(T_1' - \{t_1\}) \cup co_H(t_1\theta|t_1)$ and $(T_2' - \{t_2\}) \cup co_H(t_1\theta|t_2)$ are equivalent: i.e. the equivalence of T_1' and T_2' is an invariant of the **while**–loop in *Equivalent*.

Obviously, if $G_H(T_1') = G_H(T_2')$ then there have to exist $t_1 \in T_1'$ and $t_2 \in T_2'$ s.t. t_1 and t_2 have a common H-instance and thus are unifiable. On the other hand, $T_1' = T_2'$ implies $G_H(T_1') = G_H(T_2')$. Therefore *Equivalent* is correct.

The termination of the algorithm follows by the same argument as in Lemma 4.14: $\tau_c(T_1')$ and $\tau_c(T_2')$ are both bounded by $\tau_c(T_1 \cup T_2)$: i.e. the maximal depth of occurrence of a non-variable term (in $T_1' \cup T_2'$) cannot increase. Moreover, *Equivalent* cannot enter an infinite loop since $G_H(T_i')$ is a proper subset of $G_H((T_i' - \{t_i\}) \cup co_H(t_1\theta|t_i), i = 1, 2$. ∎

7 Evaluating clauses

As already mentioned in Section 4, for many intended applications we would like to determine algorithmically whether an arbitrary given clause is true or false w.r.t. to an interpretation that is represented by a finite set of atoms. In Section 6 we have seen that H-subsumption allows one to test atomic model representations for equivalence. In this section we show that H-subsumption also is the key concept for an algorithmic evaluation of clauses.

LEMMA 7.1
Let $\mathcal{A}$ be a finite set of atoms, C a positive clause and H a Herbrand universe. Then C is true in $INT_H(\mathcal{A})$ iff $\mathcal{A} \leq_{sH} \{C\}$.

PROOF. C is true in $INT_H(\mathcal{A})$ iff all H-instances of C are true in $INT_H(\mathcal{A})$. But a positive ground clause C' is true in the Herbrand model $INT_H(\mathcal{A})$ iff $P' \in INT_H(\mathcal{A})$ for some atom P' in C'. By definition of $INT_H(\mathcal{A})$ P' is an H-instance of some atom $P \in \mathcal{A}$. In other words, C is true in $INT_H(\mathcal{A})$ iff for all H-instances C' of C there is some $P \in \mathcal{A}$ s.t. $P \leq_{sub} C'$. The last property is obviously equivalent to $\mathcal{A} \leq_{sH} \{C\}$. ∎

To evaluate non-positive clauses we have to employ hyperresolution in addition to H-subsumption tests.

LEMMA 7.2
Let $\mathcal{A}$ be a finite set of atoms, C a non-positive clause and H a Herbrand universe. Then C is true in $INT_H(\mathcal{A})$ iff for all clash resolvents $E \in \rho_H(\mathcal{A} \cup \{C\})$ we have $\mathcal{A} \leq_{sH} \{E\}$. (Observe that this includes the case where $\rho_H(\mathcal{A} \cup \{C\})$ is empty.)

PROOF. W.l.o.g. let $C = C^+ \vee C^-$ where $LIT(C^+) = C_+$ and $LIT(C^-) = C_-$. Let $C^- = \neg Q_1 \vee \ldots \vee \neg Q_m$. Remember that there exists a clash resolvent over $\mathcal{A} \cup \{C\}$ (i.e. $\rho_H(\mathcal{A} \cup \{C\}) \neq \emptyset$) iff there exists a substitution θ and a sequence $(P_1, \ldots, P_m)$ of variants of elements in $\mathcal{A}$ s.t. $P_i\theta = Q_i\theta$ for $1 \leq i \leq m$. (We assume all P_i and C to be pairwise variable disjoint.)

(1) We first consider the case where there are no clash resolvents: Then for all H-substitutions γ and sequences $(P_1, \ldots, P_m)$ of variants of atoms in $\mathcal{A}$ we have $P_i \not\leq_{sub} Q_i\gamma$ for some i. This implies that there must be an i s.t. $\mathcal{A} \not\leq_{sub} \{Q_i\gamma\}$. Consequently $Q_i\gamma \notin INT_H(\mathcal{A})$, i.e. $\neg Q_i\gamma$ is true in $INT_H(\mathcal{A})$. We conclude that $C^-\gamma$, and thus also $C\gamma$, is true in $INT_H(\mathcal{A})$. Summarizing, the non-existence of clash resolvents over $\mathcal{A} \cup \{C\}$ implies that all H-instances of C are true in $INT_H(\mathcal{A})$. But this in turn implies that C itself is true in $INT_H(\mathcal{A})$.

(2) We assume that there is a clash resolvent $E \in \rho_H(\mathcal{A} \cup \{C\})$.

(2a) Assume that for all clash resolvents E we have $\mathcal{A} \leq_{sH} \{E\}$. We show that C must be true in $INT_H(\mathcal{A})$. Assume, to the contrary that C is false in $INT_H(\mathcal{A})$. Then some H-instance $C\gamma$ is false, too. In particular, $C^-\gamma = (\neg Q_1 \vee \ldots \vee \neg Q_m)\gamma$ is false in $INT_H(\mathcal{A})$. This implies that $Q_i\gamma \in INT_H(\mathcal{A})$ for all $1 \leq i \leq m$. Therefore there exists a sequence $(P_1, \ldots, P_m)$ of variants of elements in $\mathcal{A}$ s.t. $\{P_1, \ldots, P_m\} \leq_{sub} \{Q_i\gamma\}$ for $1 \leq i \leq m$. Thus there exists a substitution σ s.t. $P_i\sigma = Q_i\gamma$ for $1 \leq i \leq m$. By the assumption that the P_i and C are pairwise variable disjoint we can combine σ and γ to a substitution θ s.t. $P_i\theta = Q_i\theta$ for $1 \leq i \leq m$. Thus $(C\theta; P_m\theta, \ldots, P_1\theta)$ is a clash sequence that defines the clash resolvent $C^+\theta$. A simple lifting argument implies that there exists a clash resolvent E over $\mathcal{A} \cup \{C\}$ s.t. $E \leq_{sub} C^+\theta$. As already mentioned we have $\mathcal{A} \leq_{sH} \{E\}$ and consequently $\mathcal{A} \leq_{sH} \{C^+\theta\}$. By Lemma 7.1 $C^+\theta$ is true in $INT_H(\mathcal{A})$. Remember that θ is defined s.t. $C^+\theta = C^+\gamma$. But we assumed $C\gamma$ and therefore also $C^+\gamma$ to be false in $INT_H(\mathcal{A})$. The condradiction implies that C is true in $INT_H(\mathcal{A})$.

(2b) Assume that there is a clash resolvent $E \in \rho_H(\mathcal{A}\cup\{C\})$ s.t. $\mathcal{A} \not\leq_{sH} \{E\}$. By definition of H-subsumption there exists an H-instance $E\gamma$ of E s.t. $\mathcal{A} \not\leq_{sH} \{E\gamma\}$. Since E is positive we conclude by Lemma 7.1 that E, and consequently also $E\gamma$, is false in $INT_H(\mathcal{A})$.
$E \in \rho_H(\mathcal{A} \cup \{C\})$ means that there is a clash sequence $(C[= C^+ \vee C^-]; P_1, \ldots, P_m)$ s.t. $E = C^+\theta$ and $P_i\theta = Q_i\theta$ for some substitution θ and $1 \leq i \leq m$: i.e. θ is the simultaneous mgu of the clash resolution. (Again, we assume the P_i and C to be pairwise variable disjoint.)
Now $E\gamma = C^+\theta\gamma$ is known to be false in $INT_H(\mathcal{A})$. Therefore $v_{INT_H(\mathcal{A})}(C\theta\gamma) = v_{INT_H(\mathcal{A})}(C^+\theta\gamma \vee C^-\theta\gamma) = v_{INT_H(\mathcal{A})}(C^-\theta\gamma)$. By definition of θ we have $C^-\theta = (\neg P_n \vee \ldots \vee \neg P_1)\theta$. Since the $P_i\theta$ are true in $INT_H(\mathcal{A})$, $(\neg P_n \vee \ldots \vee \neg P_1)\theta$ and all its instances must be false in $INT_H(\mathcal{A})$. In particular $v_{INT_H(\mathcal{A})}(C^-\theta\gamma) = $ false. Therefore $v_{INT_H(\mathcal{A})}(C\theta\gamma) = $ false and consequently C itself is false in $INT_H(\mathcal{A})$.

∎

Combining the lemmata above we arrive at an evalution algorithm for arbitrary clauses w.r.t. any ARM.

THEOREM 7.3
Let $\mathcal{A}$ be finite a set of atoms, H a Herbrand universe and C a clause. It is decidable whether C is true or false in $INT_H(\mathcal{A})$.

PROOF. If C is positive then, by Lemma 7.1, we only need to test whether $\mathcal{A} \leq_{sH} \{C\}$. Otherwise compute $\mathcal{E} = \rho_H(\mathcal{A} \cup \{C\})$, i.e. the set of immediate clash resolvents over $\mathcal{A} \cup \{C\}$ (which is finite by definition). By Lemma 7.1, C is true in $INT_H(\mathcal{A})$ iff $\mathcal{A}$ H-subsumes $\mathcal{E}$. (Observe that this includes the case where $\mathcal{E} = \emptyset$.) But, by Theorem 6.6, H-subsumption is decidable. Therefore we can effectively evaluate C w.r.t. $INT_H(\mathcal{A})$. ∎

Evaluation of clauses w.r.t. model representations is the central technique in model-based resolution, particularly in semantic clash resolution [18]. Typically, the models used for this purpose are finite, thus admitting straightforward algorithms for the truth evaluation of clauses. The results of this section show that efficient and simple truth evaluation is also possible w.r.t. infinite models (to be precise: w.r.t. representations of infinite models). Therefore the algorithm developed above could be applied within semantic clash resolution, extending its range of applications to Herbrand models that are not just settings in the sense of [13]. Such representations of Herbrand models could be obtained by the procedure defined in Section 5. Thus hyperresolution may be used for model construction as well as for clause evaluation.

8 Extracting finite models

In the previous sections we have demonstrated that not only (finite) multiplication tables but also atomic representations of infinite Herbrand models enjoy properties that make them suitable tools for various applications. We have also shown that for the classes PVD$_+$ and OCC1N$_+$ we can algorithmically construct such model representations. Still, it would be interesting to know whether these classes are finitely controllable, i.e. whether there is a model with finite domain for every satisfiable clause set in these classes. (See [4] for a legacy of model theoretic results on finite controllability of classes of first-order formulae.) The results of this section imply a positive answer to this question.

However, from the computer science point of view, we are not only interested in finite controllability itself but strive for feasible and simple algorithms for the construction of finite models. In particular, we want to avoid exhaustive search up to some fixed limit for the cardinality of the models (as, implictly, suggested by [4]) and also do not want to introduce for this purpose equational reasoning methods on the object level (as done, for example, in [1] and [21]). Rather, we directly want to use the data generated by terminating hyperresolution procedures. To this aim we present a backtracking free algorithm that 'extracts' from any finite set of linear atoms $\mathcal{A}$ a finite model that is equivalent to $INT_H(\mathcal{A})$ (w.r.t. any H containing $H(\mathcal{A})$). This is essentially achieved by truth value preserving projections of the Herbrand base into finite subsets of it.

In order to specify the domain of our finite models we need the orthogonal extensions of sets of terms as introduced in Section 4.

DEFINITION 8.1

Let $\mathcal{A}$ be a linear atomic representation of a Herbrand model w.r.t. the Herbrand universe H (containing $H(\mathcal{A})$). We introduce an additional constant d not occurring in H. Let γ_d be the substitution that assigns d to all occurring variables; i.e. $\forall x \in V(\mathcal{A}) : \gamma_d(x) = d$. Moreover, let $T'(\mathcal{A})$ be an orthogonal extension of the set $T(A)$ of all non-variable terms that occur in some atom in $\mathcal{A}$. Then

$$D_{\mathcal{A}} = \{d\} \cup \{t\gamma_d \mid t \in T'(\mathcal{A})\}.$$

(d is intended to represent all ground terms that are not represented by any term in $T'(\mathcal{A})$.)

$D_{\mathcal{A}'}$ will serve as domain of a finite model that assigns the same truth values to atoms and clauses as $INT_H(\mathcal{A})$.

EXAMPLE 8.2
Let

$$\mathcal{A} = \{P(f(x,g(y)),g(z)),\ P(g(f(x,a)),y),\ P(f(g(x),y),g(z))\}.$$

Since

$$T(\mathcal{A}) = \{f(x,g(y)),\ g(x),\ g(f(x,a)),\ f(x,a),\ a,\ f(g(x),y)\}$$

we may refer to Example 4.4 to obtain the orthogonal extension

$$\begin{aligned} T'(\mathcal{A}) = \{&a,\ g(f(x,a)),\ g(a),\ g(g(x)),\ g(f(x,g(y))),\ g(f(x,f(y,z))),\\ &f(a,g(x)),\ f(f(x,y),a),\ f(a,a),\ f(g(u),g(y)),\ f(f(x,y),g(z)),\\ &f(g(x),a),\ f(g(x),f(y,z))\}. \end{aligned}$$

Consequently we have

$$\begin{aligned} D_{\mathcal{A}} = \{&d,\ a,\ g(f(d,a)),\ g(a),\ g(g(d)),\ g(f(d,g(d))),\ g(f(d,f(d,d))),\\ &f(a,g(d)),\ f(f(d,d),a),\ f(a,a),\ f(g(d),g(d)),\ f(f(d,d),g(d)),\\ &f(g(d),a),\ f(g(d),f(d,d))\}. \end{aligned}$$

By Lemma 4.14 we can always compute an orthogonal extension of the set $T(\mathcal{A})$ of terms occurring in a linear atomic representation $\mathcal{A}$ of the Herbrand model $INT_H(\mathcal{A})$. This allows us to define the following function that assigns some element of $D_{\mathcal{A}}$ to each term of the corresponding Herbrand universe.

DEFINITION 8.3
Let $\mathcal{A}$ be a linear atomic representation of a Herbrand model of a clause set $\mathcal{C}$. Let $H = H(\mathcal{C})$ and let $T'(\mathcal{A})$ be an orthogonal extension of $T(\mathcal{A})$ w.r.t. H. Then for each $t \in H \cup \mathcal{D}_{\mathcal{A}}$ we define $\Phi_{\mathcal{A}}(t)$ as follows:

(i) If t is an instance of some $s \in T'(\mathcal{A})$ then $\Phi_{\mathcal{A}}(t) = s\gamma_d$ (where γ_d is defined as in Definition 8.1).
(ii) Otherwise, let $\Phi_{\mathcal{A}}(t) = d$.

The definition of $\Phi_{\mathcal{A}}$ enables us to specify concisely the interpretation of the function symbols in the finite model. We are now in the position to present the complete specification of a finite model corresponding to an $INT_H(\mathcal{A})$.

DEFINITION 8.4
Let $\mathcal{A}$ be a linear atomic representation of a Herbrand model of some clause set $\mathcal{C}$. Then the (finite) interpretation $\mathcal{FM}_{\mathcal{A}} = \langle D, \varphi\rangle$ is defined as follows:

(i) $D = D_{\mathcal{A}}$.
(ii) $\varphi(c) = c$ for all constants $\in D_{\mathcal{A}}$. For all other constants occurring in $\mathcal{C}$ we define $\varphi(c') = d$.
(iii) $\varphi(f)(t_1,\ldots,t_n) = \Phi_{\mathcal{A}}(f(t_1,\ldots,t_n))$ for all n-ary function symbols f occurring in $\mathcal{C}$.
(iv) $\langle t_1,\ldots,t_n\rangle \in \varphi(P)$ iff $P(t_1,\ldots,t_n)$ is an instance of some $A \in \mathcal{A}$. for all n-ary predicate symbols P occurring in $\mathcal{C}$ and all terms $t_1,\ldots t_n \in D_{\mathcal{A}}$.

Of course, instead of using the set of terms $D_{\mathcal{A}}$ as domain of discourse, we could map $D_{\mathcal{A}}$ bijectively into any domain of the same cardinality and define φ accordingly.

EXAMPLE 8.5
Again, we take up the above example where $\mathcal{A} = \{P(f(x, g(y)), g(z)),\ P(g(f(x, a)), y),\ P(f(g(x), y), g(z))\}$.

To be able to present the interpretation of the function symbols g and f concisely we map the fourteen elements of $D_{\mathcal{A}}$ into $D = \{0, 1, \ldots, 13\}$ as follows:

$$d = 0,\ a = 1,\ g(f(d,a)) = 2,\ g(a) = 3,\ g(g(d)) = 4,$$
$$g(f(d, g(d))) = 5,\ g(f(d, f(d,d))) = 6, f(a, g(d)) = 7,$$
$$f(f(d,d), a) = 8,\ f(a,a) = 9,\ f(g(d), g(d)) = 10,$$
$$f(f(d,d), g(d)) = 11,\ f(g(d), a) = 12,\ f(g(d), f(d,d)) = 13.$$

Definition 8.4 leads us thus to the finite model $\mathcal{FM}'_{\mathcal{A}} = \langle D, \varphi \rangle$, where $\varphi(d) = 0$, $\varphi(a) = 1$ and $\varphi(g)$ and $\varphi(f)$ are defined as follows:

g	0	1	2	3	4	5	6	7	8	9	10	11	12	13
$\varphi(g)$	0	3	4	4	4	4	4	5	2	2	5	5	2	6

$\varphi(f)$	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	9	7	7	7	7	7	0	0	0	0	0	0	0
2	0	12	10	10	10	10	10	13	13	13	13	13	13	13
3	0	12	10	10	10	10	10	13	13	13	13	13	13	13
4	0	12	10	10	10	10	10	13	13	13	13	13	13	13
5	0	12	10	10	10	10	10	13	13	13	13	13	13	13
6	0	12	10	10	10	10	10	13	13	13	13	13	13	13
7	0	8	11	11	11	11	11	0	0	0	0	0	0	0
8	0	8	11	11	11	11	11	0	0	0	0	0	0	0
9	0	8	11	11	11	11	11	0	0	0	0	0	0	0
10	0	8	11	11	11	11	11	0	0	0	0	0	0	0
11	0	8	11	11	11	11	11	0	0	0	0	0	0	0
12	0	8	11	11	11	11	11	0	0	0	0	0	0	0
13	0	8	11	11	11	11	11	0	0	0	0	0	0	0

We remark that if there is an H–model $\mathcal{M}$ for some clause set C, s.t. $\mathcal{M} = \{A \mid v_{\mathcal{M}}(A) = \mathbf{true}\}$ is finite then we have a simple subcase of our approach: in that case $\mathcal{M}$ itself may be conceived as a finite set of linear atoms. Moreover, the set $T(\mathcal{M})$ of terms occurring in $\mathcal{M}$ is an orthogonal representation of itself. Therefore $\mathcal{FM}_{\mathcal{M}}$ is a model of cardinality $|T(\mathcal{M}) + 1|$ (the additional element d represents all ground terms that are not in $T(\mathcal{M})$). Observe that, in general, this model is not minimal w.r.t. the cardinality of the domain.

The main lemma to justify our construction is the following:

LEMMA 8.6
For all ground terms t: $v_{\mathcal{FM}_{\mathcal{A}}}(t) = \Phi_{\mathcal{A}}(t)$.

PROOF. We proceed by induction on the term depth of t.

If $\tau(t) = 0$, i.e. t is a constant, then, by definition of v as an evaluation function we have $v(t) = \varphi(t)$. But for all constants c: $\varphi(c) = c$, or (if c is not in $D_{\mathcal{A}}$) $\varphi(c) = d$. Thus, by Definition 8.4, $v(t) = \Phi_{\mathcal{A}}(t)$.

We have the following induction hypothesis:

(IH) $v_{\mathcal{FM}_{\mathcal{A}}}(t) = \Phi_{\mathcal{A}}(t)$ for all t s.t. $\tau(t) \leq n$

Let t' be of depth $n+1$, thus $t' = f(t_1, \ldots, t_k)$, where $\tau(t_i) \leq n$ for $1 \leq i \leq k$. Thus, by definition of v: $v(t') = v(f(t_1, \ldots, t_k)) = \varphi(f)(v(t_1), \ldots, v(t_k))$. Therefore, by (IH), $v(t') = \varphi(f)(\Phi_{\mathcal{A}}(t_1), \ldots, \Phi_{\mathcal{A}}(t_k))$. By definition of $\varphi(f)$ we obtain $v(t) = \Phi_{\mathcal{A}}(f(\Phi_{\mathcal{A}}(t_1), \ldots, \Phi_{\mathcal{A}}(t_k))$. Now observe that $\Phi_{\mathcal{A}}(f(\Phi_{\mathcal{A}}(t_1), \ldots, \Phi_{\mathcal{A}}(t_k)) = \Phi_{\mathcal{A}}(f(t_1, \ldots, t_k))$. We conclude that $v_{\mathcal{FM}_{\mathcal{A}}}(t) = \Phi_{\mathcal{A}}(t)$ for all terms t. ∎

Given Lemma 8.6 it is an easy task to prove:

THEOREM 8.7
If $\mathcal{A}$ is a linear atomic representation of a Herbrand model of some clause set $\mathcal{C}$, then $\mathcal{FM}_{\mathcal{A}}$ is a finite model for $\mathcal{C}$.

PROOF. Clearly, $\mathcal{FM}_{\mathcal{A}} = \langle D_{\mathcal{A}}, \varphi \rangle$ is a finite interpretation for $\mathcal{C}$. By Herbrand's theorem it suffices to show that for any ground atom A

$$v_{INT_H(\mathcal{A})}(A) = \mathbf{true} \Longleftrightarrow v_{\mathcal{FM}_{\mathcal{A}}}(A) = \mathbf{true},$$

where $H = H(\mathcal{C})$. Let $A = P(t_1, \ldots t_n)$ be in $INT_H(\mathcal{A})$, (i.e. A is true in $INT_H(\mathcal{A})$). By Lemma 8.6 $v_{\mathcal{FM}_{\mathcal{A}}}(A) = \mathbf{true}$ iff $\langle \Phi_{\mathcal{A}}(t_1), \ldots \Phi_{\mathcal{A}}(t_n) \rangle \in \varphi(P)$. But since $\Phi_{\mathcal{A}}$ is based on an orthogonal extension of $T(\mathcal{A})$ we have $P(\Phi_{\mathcal{A}}(t_1), \ldots \Phi_{\mathcal{A}}(t_n)) = A'\gamma_d$ for some $A' \in \mathcal{A}$. It follows that A is true in $\mathcal{FM}_{\mathcal{A}}$, too.

On the other hand assume that $\langle \Phi_{\mathcal{A}}(t_1), \ldots \Phi_{\mathcal{A}}(t_n) \rangle \in \varphi(P)$. It follows by the definition of $\Phi_{\mathcal{A}}$ that $A = P(t_1, \ldots t_n)$ is an instance of some $A' \in \mathcal{A}$ s.t. $P(\Phi_{\mathcal{A}}(t_1), \ldots \Phi_{\mathcal{A}}(t_n)) = A'\gamma_d$. In other words: A is true in $INT_H(\mathcal{A})$. ∎

The following proposition demonstrates that, using our approach for finite model building (i.e. projecting the infinite domain of a Herbrand model into a finite set) one cannot go essentially beyond classes that allow one to construct linear ARMs.

PROPOSITION 8.8
Let $\mathcal{A} = \{P(x, f(x))\}$ be a representation of the Herbrand model $INT_H(\mathcal{A})$ w.r.t. $H = H(\mathcal{A})$. Then there is no finite interpretation $\mathcal{M}_D$ of (the Herbrand base of) $\mathcal{A}$ s.t. for all ground atoms P:

$$v_{\mathcal{M}_D}(P) = \mathbf{true} \Longleftrightarrow v_{INT_H(\mathcal{A})}(P) = \mathbf{true}.$$

PROOF. Let $\mathcal{M}_D = \langle D, \varphi_D \rangle$ be any finite interpretation of $\mathbf{HB}(\mathcal{A})$. By definition of the evaluation function we have

$$v_{\mathcal{M}_D}(f^n(a)) = (\varphi_D(f))^n(\varphi_D(a)),$$

where a is the only constant in H. Since $\varphi_D(f)$ can only take finitely many different values there must be some $n, m \geq 1$, s.t. $n \neq m$ but

$$v_{\mathcal{M}_D}(f^n(a)) = v_{\mathcal{M}_D}(f^m(a)).$$

This implies that

$$v_{\mathcal{M}_D}(P(f^n(a), f^{n+1}(a)) = v_{\mathcal{M}_D}(P(f^m(a), f^{n+1}(a)).$$

On the other hand, by definition of $INT_H(\mathcal{A})$,

$$v_{INT_H(\mathcal{A})}(P(f^k(a), f^{l+1}(a)) = \mathbf{true} \qquad \text{iff} \qquad k = l.$$

It follows that

$$v_{\mathcal{M}_D}(P(f^m(a), f^{n+1}(a)) \neq v_{INT_H(\mathcal{A})}(P(f^m(a), f^{n+1}(a)).$$

Therefore $v_{INT_H(\mathcal{A})}(P)$ and $v_{\mathcal{M}_D}(P)$ are different for some ground atoms P. ∎

9 Conclusions

In this paper we aimed at the construction of models for certain sets of clauses using the set of all hyperresolvents, given that hyperresolution terminates without deriving $\square$. Moreover, we have demonstrated that the resulting representations of Herbrand models share some desireable features with standard representations of finite models. In particular, we can decide whether two given representations represent the same model and can recursively evaluate arbitrary clauses w.r.t. to the represented model. Especially the last mentioned feature indicates that our methods can be applied to pruning proof search by generating countermodels.

As concrete examples of our approach to model building we investigated two classes of clause sets that were demonstrated to be decidable by hyperresolution in [7]. The process of model (representation) construction proceeds in two steps: first hyperresolution is employed to arrive at a finite set of atoms that represents a description of a Herbrand model. In a second step we may extract from this set of atoms a full representation of a model with finite domain. We emphasize that the second step can be performed on every stable set of this specific syntax type, no matter how this set was obtained. In this sense, step 2 is independent of step 1.

By the methods developed in this paper we obtain finite models for all satisfiable sets of clauses contained in one of the decidable classes in Section 3. This might suggest that termination of hyperresolution without deriving contradiction implies the finite model property — a conjecture formulated in [6]. However, this conjecture is easily falsified by considering the following set of clauses (communicated by M. Baaz):

$$\mathcal{C} = \{P(x,x), \neg P(f(x), f(y)) \vee P(x,y), \neg P(c, f(x))\}.$$

Hyperresolution terminates on $\mathcal{C}$, giving the atomic model representation $\{P(x,x)\}$. But $\mathcal{C}$ does not have finite models. This example also shows that our methods for the construction of model representations (Section 5) and evaluation of clauses w.r.t. models (Section 7) surpass the range of finitely controllable classes.

We consider the work of T. Tammet and the presented results just as a starting point in the more general project to extract (information about) models from termination sets of resolution provers. A direction for future research arises if we do not only consider hyperresolution but, also ordering strategies and other resolution refinements, possibly combined with certain methods for equality reasoning.

References

[1] R. Caferra and N. Zabel. Extending resolution for model construction. In *Logics in AI (JELIA '90)*, pp. 153–169. Springer Verlag, 1991. LNCS 478.

[2] R. Caferra and N. Zabel. A method for simultaneous search for refutations and models by equational constraint solving. *J. Symbolic Computation*, 13, 613–641, 1992.

[3] C.-L. Chang and R. C.-T. Lee. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, New York and London, 1973.

[4] B. Dreben and W. D. Goldfarb. *The Decision Problem*. Addison-Wesley, 1979.

[5] C. Fermüller. Deciding some Horn clause sets by resolution. In *Yearbook of the Kurt-Gödel-Society 1989*, pp. 60–73, Vienna, 1990.

[6] C. Fermüller and A. Leitsch. Model building by resolution. In *Computer Science Logic (CSL'92)*, pp. 134–148. San Miniato, Italy, 1993. Springer Verlag. LNCS 702.

[7] C. Fermüller, A. Leitsch, T. Tammet, and N. Zamov. *Resolution Methods for the Decision Problem*. Springer Verlag, 1993. LNAI 679.

[8] G. Gottlob. Complexity results on nonground Herbrand interpretations. In preparation, 1994.

[9] J.-M. Hullot. Canonical forms and unification. In *5th Conference on Automated Deduction*, pp. 318–334. Springer Verlag, 1980. LNCS 87.

[10] W. H. Joyner. Resolution strategies as decision procedures. *J. Association of Computing Machinery*, **23**, 398–417, 1976.

[11] A. Leitsch. Deciding Horn classes by hyperresolution. In *Computer Science Logic (CSL'89)*, pp. 225–241, Kaiserslautern, 1990. Springer Verlag. LNCS 440.

[12] A. Leitsch. Deciding clause classes by semantic clash resolution. *Fundamenta Informaticae*, **18**, 163–182, 1993.

[13] D. Loveland. *Automated Theorem Proving — A Logical Basis*. North Holland, 1978.

[14] R. Manthey and F. Bry. Satchmo: A theorem prover implemented in Prolog. In *9th Conference on Automated Deduction*, pp. 415–434. Springer Verlag, 1988. LNCS 310.

[15] A. Martelli and U. Montenari. An efficient unification algorithm. *ACM Transactions on Programming Languages and Systems*, **4**, 258–282, 1982.

[16] W. McCune. *Otter 2.0 Users Guide*. Argonne National Laboratory, Argonne, 1990.

[17] H. Noll. A note on resolution: How to get rid of factoring without losing completeness. In *5th Conference on Automated Deduction*, pp. 250–263. Springer Verlag, 1980. LNCS 87.

[18] J. R. Slagle. Automatic theorem proving with renamable and semantic resolution. *J. Association of Computing Machinery*, **19**, 496–516, 1972.

[19] J. Slaney. Finder (finite domain enumerator): Notes and guide. Technical report TR-ARP-1/92, Australian National University Automated Reasoning Project, Canberra, 1992.

[20] J. Slaney. Scott: A model-guided theorem prover. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI '93)*, vol. 1, pp. 109–114. Morgan Kaufmann, 1993.

[21] T. Tammet. Using resolution for deciding solvable classes and building finite models. In *Baltic Computer Science*, pp. 33–64. Springer Verlag, 1991. LNCS 502.

[22] T. Tammet. *Resolution Method for Decision Problems and Finite Model Building*. PhD thesis, Department of Computer Science, Chalmers University of Technology, Chalmers/Göteborg, 1992.

[23] S. Winker. Generation and verification of finite models and counterexamples using an automated theorem prover answering two open questions. *J. Association of Computing Machinery*, **29**, 273–284, 1982.

Received 24 March 1994