ON THE COMPLEXITY OF HERBRAND's THEOREM

by

Alon Itai and Johann Makowsky

Technical Report #243

May 1982

* Partially sponsored by Swiss National Science Foundation
  Grant No. 82.820.80.
On the Complexity of Herbrand's Theorem

Alon Itai
Johann Makowsky

Computer Science Department
Technion - IIT, Haifa, Israel

Keywords: Automated theorem proving, complexity, Herbrand's Theorem, Horn formulas.

1. Introduction

Horn formulas and Herbrand's Theorem appear frequently in the applications of logic to Computer Science: automated theorem proving [CL, Rob], specification of abstract data types and dependencies of relational data base schemes [ADJ, Fa, MM]. The aim of this paper is to study the complexity involved in Herbrand's Theorem in general and in particular in the simplest case where no function symbols are involved and the quantifier-free part is a Horn formula. Our main result is that there is a constant $d > 1$ such that for every $n$ there is an unsatisfiable universal Horn formula $\forall x H_n(x)$ of length $n$, but to see this at least $d^n$ instances of $H_n$ are needed. If function symbols are allowed there is no recursive bound on the number of instances, however, this number determines the length of the formulas.

1 Partially supported by Swiss National Science Foundation Grant No. 82.820.80.
The paper is organized as follows: In section 2 we present the results based on several lemmas, each of which is proved and elaborated in a separate section. Section 3 spells out some facts about the unification algorithm. Section 4 gives a linear algorithm for testing satisfiability of propositional Horn formulas. In section 5 we give some open problems and relate our work to existing results in the literature.

Besides standard knowledge and the theorem quoted in [Cl.M] the paper is self contained.

2. The Main Result

Let \( \varphi \) be a first-order formula over a set \( \mathcal{L} \) of functions, relations and constant symbols. Let

\[
\forall x_1 \ldots x_n(\varphi) \text{Sk}(x_1, \ldots, x_n(\varphi))
\]

be its canonical Skolem normal form with \( \text{Sk}(\varphi) \) quantifier-free and

\[
\text{skf}(\varphi) = \{ F_1, \ldots, F_S(\varphi) \}
\]

the set of Skolem functions in \( \text{Sk}(\varphi) \). Now Herbrand's Theorem is the following:

**Theorem H** For every first order sentence \( \varphi \) over \( \mathcal{L} \) there is a \( k \in \mathbb{N} \) and there are constant terms \( t_j^{(i \leq n(\varphi), j \leq k)} \) over \( L \cup \text{skf}(\varphi) \) such that if \( \varphi \) is unsatisfiable then

\[
\psi = \bigwedge_{j=1}^{k} \text{Sk}(t_1^{(i)}, \ldots, t_k^{(i)})
\]

is unsatisfiable. We shall call such a \( \bigwedge_{j=1}^{k} \text{Sk}(t_1^{(i)}, \ldots, t_k^{(i)}) \) an *Herbrand expansion* of \( \varphi \) of size \( k \) and (binary) length \( l(\psi) \). (If \( \varphi \) is satisfiable then any instantiation of \( \text{Sk}(\varphi) \) is an Herbrand expansion.)

Let \( HL(\varphi) \), the *Herbrand length* of \( \varphi \) be the minimum of the binary length of all Herbrand expansions of \( \varphi \). (Hence if \( \varphi \) is satisfiable we put \( HL(\varphi) = l(\text{Sk}(\varphi)) \).)
Let $HS(\varphi)$, the Herbrand size of $\varphi$, be the minimum size of all Herbrand expansions of $\varphi$. (Hence $HS(\varphi)=1$ for all satisfiable $\varphi$.)

Let $\Sigma$ be a recursive set of first-order formulas, $f: \Sigma \to \mathbb{N}$ a function from $\Sigma$ to the natural numbers $\mathbb{N}$. $f$ is recursively bounded if there is a recursive function $g: \Sigma \to \mathbb{N}$ such that

$$f(\varphi) \leq g(\varphi) \text{ for every } \varphi \in \Sigma.$$ 

As an immediate consequence of Church's result that the valid first order formulas are not recursive, we get

**Theorem C** $HL(\varphi)$ is not recursively bounded.

This theorem has several refined versions for various subset $\Sigma$ of first order formulas with restriction on their quantifier structure or of the language $L$

**Theorem K** $HL(\varphi)$ is not recursively bounded even if restricted to $\Sigma$ with

$\Sigma_1$ - the set of first order formulas (without equality) over a single binary predicate symbol.

$\Sigma_2$ - the set of universal Horn formulas with equality, a single binary function symbol and a finite number of constant symbols.

For $\Sigma_1$ this follows from a theorem by Kalmar (1932) that satisfiability of $\Sigma_1$ formulas is undecidable. For $\Sigma_2$ this follows from the unsolvability of the word problem in groups and semigroups [Lew, Sh]. We wish to extend these results to $HS(\varphi)$.

Let $\Sigma$ be a recursive set of first order sentences, $\Sigma$ is *satisfiability decidable* ($s$-decidable) if the set of satisfiable sentences in $\Sigma$ is recursive.

**Theorem 1** A recursive set $\Sigma$ of first order sentences is $s$-decidable iff for $HS$ is recursively bounded on $\Sigma$.

The corresponding statement for $HL(\varphi)$ is trivial and the reduction necessary to prove Theorem 1 is contained in the following lemma:
Lemma 2 For a recursive set $\Sigma$ of first order sentences $HS$ restricted to $\Sigma$ is recursively bounded if $HL$ restricted to $\Sigma$ is recursively bounded.

The lemma will be proved in Section 3, via a close analysis of the unification algorithm for first order logic.

In the case where $\Sigma$ is the set of universal sentences without function symbols but with constant symbols $c_1, \ldots, c_d$ there are at most $d^n$ substitution instances for $\varphi = \forall x_1 \cdots x_n B$ (B quantifier-free). Hence, $HS(\varphi) \leq d^n$.

The same applies to $\Sigma_3$ the set of sentences of the form

$$\varphi = \exists x_1 \cdots x_m \forall x_{m+1} \cdots x_{m+n} B$$

(B quantifier free)

since $skf(\varphi)$ contains exactly $m$ constant symbols.

Our next theorem shows that this is the best possible:

Theorem 3 There is a finite set $L$ of relation symbols and a number $c > 1$ such that for $\Sigma_3$ the set of $\exists \forall L$-sentences $HS(\varphi) \geq c^{l(\varphi)}$ for infinitely many $\varphi \in \Sigma_3$.

Theorem 3 will follow from Theorem 5 below, which states the same result even for the case where the matrix $B$ of the $\exists \forall$-formula is a (quantifier-free) Horn formula.

In many areas of Computer Science (such as, automated theorem proving, specification of abstract data-types and data-base theory [Rob, MM, ADJ, FA]) it is generally believed that Horn formulas are particularly well-suited for automated processing. Arguments supporting this are:

1. Horn formulas can be given a procedural interpretation as non-deterministic programs, cf. [Ko].

2. Even simple resolution gives a quadratic satisfiability test for sets of propositional Horn formulas.

Now Theorem 5 together with Lemma 2 show that the fact that $B$ is a Horn formula does not affect $HS$ as $HL$ if they are not recursively bounded.
Our main result is the improvement of Theorem 3 in the case where $B$ is a Horn matrix. Our proof splits into two parts:

**Proposition 4** There is a linear decision procedure for the satisfiability of Horn formulas.

This will be proved in section 4. Then we apply the following theorem:

**Theorem CLM** There is a constant $d > 1$ such that the satisfiability problem for universal Horn formulas with relation symbols and constants cannot be solved in time $d^l$ (where $l$ is the (binary) length of the formula).

Proposition 4 and Theorem CLM immediately give us:

**Theorem 5** There is a constant $d > 1$ such that $HS(\varphi) > d^{l(\varphi)}$ for arbitrary large $\varphi \in \Sigma$, with $\Sigma$ the class of universal Horn formulas with only relation and constant symbols.

To obtain Theorem CLM (and Theorem 5) the number of constants and the "arity" of the relation symbols in $\varphi$ are allowed to grow with $\varphi$. In the final section we discuss improvements and limitations of this.

### 3. A closer look at unification

The aim of this section is to spell out several facts about unification which are implicit in [CL] and [DG], and to prove Lemma 2. We leave the treatment informal in order to reduce the technical definitions to a minimum. Notation is standard as in [CL].

We denote by $C_1(\vec{x})$, $C_2(\vec{y})$ clauses of predicate calculus, $\vec{x}$, $\vec{y}$ vectors of variables, $\vec{t}_i$ vectors of terms, $\sigma, \tau, \mu$ substitutions. A substitution unifies two sets of terms, if they become equal after the simultaneous execution of this substitution. A *most general unifier* (mgu) for these two sets is a unifier such that any other unifier for it can be produced by a further substitution.
Let $C_1(\bar{t}_1), C_2(\bar{t}_2)$ be two clauses over terms $\bar{t}_1, \bar{t}_2$. We denote by $\text{res}(C_1(\bar{t}_1), C_2(\bar{t}_2))$ the set of clauses $D(\bar{t}_1, \bar{t}_2)$ which can be obtained from $C_1(\bar{t}_1), C_2(\bar{t}_2)$ by one resolution step.

**Lemma 3.1** Let $C_1(\bar{x}), C_2(\bar{y})$ be clauses with disjoint variables $\bar{x}, \bar{y}$. Let $\bar{t}_1, \bar{t}_2$ be vectors of constant terms and $D(\bar{t}_1, \bar{t}_2) \subseteq \text{res}(C_1(\bar{t}_1), C_2(\bar{t}_2))$. Then

(i) $C_1(\bar{x})$ and $C_2(\bar{y})$ are unifiable.

(ii) There is a mgu $\mu$ for $C_1(\bar{x})$ and $C_2(\bar{y})$ and a substitution $\sigma$ and $D_0 \subseteq \text{res}(C_1(\bar{x}) \mu, C_2(\bar{y}) \mu)$ such that $D = D_0 \sigma$.

**Proof:** W.l.o.g.

$$C_1(\bar{x}) = p_1(\bar{x}) \lor C_{11}(\bar{x})$$
$$C_2(\bar{y}) = \neg p_2(\bar{x}) \lor C_{21}(\bar{x})$$

where $p_1$ and $p_2$ are atomic formulas with the same predicate letter. Now there are substitutions $\tau_1$ and $\tau_2$ such that $\tau_1$ does not affect $\bar{y}$ and $\tau_2$ does not affect $\bar{x}$ and $C_1(\bar{x}) | \tau_1 = C_1(\bar{t}_1)$ and $C_2(\bar{y}) | \tau_2 = C_2(\bar{t}_2)$. Since $C_1(\bar{x}) | \tau_1$ is variable-free

$$C_1(\bar{x}) | \tau_1 \tau_2 = C_1(\bar{x}) | \tau_1 = C_1(\bar{t}_1) = p_1(\bar{t}_1) \lor C_{11}(\bar{t}_1)$$

Since $\tau_1$ does not change $\bar{y}$, $C_2(\bar{y}) | \tau_1 = C_2(\bar{y})$ and

$$C_2(\bar{y}) | \tau_1 \tau_2 = C_2(\bar{y}) | \tau_2 = C_2(\bar{t}_2) = \neg p_2(\bar{t}_2) \lor C_{21}(\bar{t}_2)$$

Thus $C_1(\bar{x})$ and $C_2(\bar{y})$ are unifiable by $\tau_1 \tau_2$. So let $\mu$ be a mgu for $\tau_1 \tau_2$. Thus there is $\sigma$ with $\mu \sigma = \tau_1 \tau_2$.

Now we put

$$D_0 = C_{11}(\bar{x}) | \mu \lor C_{21}(\bar{y}) | \mu$$
$$D = C_{11}(\bar{x}) | \tau_1 \tau_2 \lor C_{21}(\bar{y}) | \tau_1 \tau_2$$

and see that $D_0 | \sigma = D$. QED

Continuing the argument we can show that

**Corollary 3.2** If $\mu_1, \mu_2$ are mgu's for $C_1(\bar{x}), C_2(\bar{y})$ then
(i) the length of $C_1(\overline{x})|_{\mu_1}$ and $C_1(\overline{x})|_{\mu_2}$ are the same.

(ii) $\mu_1$ and $\mu_2$ are the same up to renaming of the variables.

Proof:

(i) There is a $\sigma$ with $C_1(\overline{x})|_{\mu_2}=C_1(\overline{x})|_{\mu_1}$. But since substitutions only increase length $l(C_1(\overline{x})|_{\mu_2})=l(C_1(\overline{x})|_{\mu_1})\geq l(C_1(\overline{x})|_{\mu_1})$ and by symmetry equality follows.

(ii) If a substitution is length preserving then it can only rename variables.

QED

Proof of Lemma 2:

Let $\varphi$ be a formula which is not satisfiable and let $\psi=\bigwedge_{f=1}^k \varphi(t_{f1}', \ldots, t_{f(n)}')$ be its Herbrand expansion, such that $\psi$ is unsatisfiable. Using, for instance, the Davis-Putnam procedure, there is a refutation tree $T$ whose depth and number of leaves is bounded by a recursive function of $k$ and the length of $\varphi$ (independently of the choice of the terms $t_f'$). From Lemma 3.1 there is a refutation-tree $T'$ such that

(i) $T'$ is isomorphic to $T$.

(ii) $T'$ uses only mgu's

Let $C(\overline{z}) \in \text{res}(C_1(\overline{x})|_{\mu}, C_2(\overline{y})|_{\mu})$ where $\mu$ is a mgu, then $|C(\overline{z})| \leq 2|C_1(\overline{x})| + |C_2(\overline{y})|.$

Consequently, the sum of the lengths of all terms of $T'$ is bounded by a recursive function of $k$ and $l(\varphi)$. If $k=HK(\varphi)$ were recursively bounded, we would have a decision procedure for satisfiability of first order formulas. QED

4. The linear algorithm for propositional Horn formulas

In this section we present a linear algorithm to decide whether a given set of Horn formulas is satisfiable. Recall that a Horn formula is a clause with at
most one non-negated atomic formula. Our proof has two stages: First we present a linear reduction to the propositional case and then we give a linear algorithm to solve the latter.

Let \( P = \{p_0, p_1, p_2, \ldots \} \) be the set of propositional variables, \( A = \{a_0, a_1, a_2, \ldots \} \) the set of atomic formulas over some countable similarity type and \( \varphi \) some quantifier-free formula over the similarity type. If the set of atomic formulas of \( \varphi \) is \( fA \) then \( \varphi \) is logically equivalent to \( \varphi^P \) obtained from \( \varphi \) by replacing each atomic formula \( a_j \) by the propositional variable \( p_j \).

However, this transformation is not satisfactory from a complexity point of view, since the maximum index of the variables might be very large (consider, for example, the case where the space depends on the maximum index). To remedy this problem, the atomic formulas \( fA \) are replaced by the propositional variables \( \{p_0, p_1, \ldots, p_r\} \), such that \( p_j \) replaces the \( j \)-th formula in the lexicographic ordering of \( fA \). To find the lexicographic index of the \( a_j \)'s we construct a binary tree corresponding to their binary representation. The lexicographic index of the \( a_j \)'s can be found by traversing the tree in infix order. The entire transformation involves two passes over \( \varphi \) and one tree traversal, thus \( O(l(\varphi)) \) time. Note also that if \( a_j \) is renamed \( p_k \) then \( |p_k| \leq |a_j| \). Thus \( |\varphi^P| \leq |\varphi| \).

We now turn to the linear algorithm to check satisfiability of a set of Horn formulas. A basic Horn formula over the propositional variables \( \{P_0, \ldots, P_n\} \) is a disjunction of literals of which at most one is positive. We wish to check satisfiability of a set \( \varphi \) of basic Horn formulas.

Any basic Horn formula can be written in one of the following forms:

\( H_1: Q_1 \text{ and } \ldots \text{ and } Q_m \rightarrow R_i \)

\( H_2: Q_j \)

where each \( Q_i \) is some \( P_h \) and \( R_i \) is either some \( P_j \) or the constant false.
A true variable is a variable which must be assigned the value true in any satisfying assignment. All variables appearing in $H_2$-formulas are true. The algorithm replaces the original $\varphi$ by equivalent sets. The new sets are derived by either deleting a basic Horn formula whose right-hand-side is a true variable, or by deleting a true variable from a the left-hand-side of an $H_1$-formula. If the entire left-hand-side is deleted then the variable on the right-hand-side becomes a true variable. $\varphi$ is unsatisfiable if $Q_0$ becomes true. The algorithm terminates successfully when no true variable appears in any $H_1$-formula. Then $\varphi$ can be satisfied by assigning the value false to all variables appearing in $H_1$-formulas.

To implement the above algorithm we should be able to do the following operations:

(i) Find the next true variable to be processed.
(ii) Delete all occurrences of a true variable.
(iii) Delete an $H_1$-formula the right-hand-side of which is a true variable.

To implement (i) we shall maintain a queue of all unprocessed true variables, such that adding and removing a single variable takes constant time.

The right-hand-side of the $H_1$-formulas are represented by a vector. As for the left-hand-side, we first sort the left-hand-side of each $H_1$-formula by index of the variables. Using bucket-sort the entire reordering requires $O(|\varphi|)$ time. Now left-hand-side of the $H_1$-formulas can be represented by a sparse matrix whose rows represent formulas and whose columns represent variables. The matrix consists of nodes, one for each occurrence of a variable, with pointers to the next variable in the formula and to the next occurrence of the variable. Pointers are maintained to the first variable of each formula and to the first and last occurrence of each variable. (This last pointer is required only to create the structure in linear time and can be discarded later.)
The number of nodes is linear in the length of $\varphi$. Since each node can be deleted at most once, and the deletion of a node requires constant time, the entire algorithm is linear.

This proves Proposition 4.

5. Improvements, open problems and comments

In the proof of Theorem CLM and hence for Theorems 3 and 5 relation symbols of unbounded arity are used. In fact, in the formula of length $O(n)$ relation symbols of arity $O(n \log n)$ are used. If one allows arity $O(n)$ relation symbols, Theorem 3 can be proved directly in the following way (as suggested by H. Lewis): Write a formula of arity $O(n)$ with an $n$-ary relation symbol $P$ and two constants 0, 1 which says:

$P(0, 0, \ldots, 0)$ is true

$P(1, 1, \ldots, 1)$ is false

and if $P(y_1, \ldots, y_n)$ is true and $y_1 \ldots y_n$ is the binary notation for $k$ and $z_1 \ldots z_n$ is the binary notation for $k+1$ then $P(z_1, \ldots, z_n)$ is true. This can be done by a formula of length $O(n)$ using coding tricks, also H. Lewis has shown that $\Omega(n^n)$ are needed to show its inconsistency.

However, it is preferable to keep the arity of the relation symbols bounded. As in descriptive geometry, where every $n$-ary relation can be represented faithfully by $n-1$ binary relations (the various projections on the planes), allowing additional constant symbols one ternary relation symbol suffices. The length of the formula obtained is quadratic in the arity $n$.

Hence we have the following problems:

**Problem 1** Is $d^n$ a lower bound for $HS(\varphi)$ if $\varphi$ contains only one fixed relation symbol of fixed arity $n$?
Problem 2 Is $d^n$ a lower bound for $HS(\varphi)$ if both the number of constant symbols, the number and arity of the relation symbols are bounded?

For a class $\Sigma$ of formulas which is decidable in $f(l(\varphi))$ time our examples always imply that $HS(\varphi)$ is bounded $O(f)$.

Problem 3 Is there a class $\Sigma$ of formulas which is decidable in $f(l(\varphi))$ but $HS(\varphi)$ is not bounded by $O(f)$?

Note that M. Wietlisbach [Wie] has shown that there are formulas in which both the Herbrand size and the resolution part are exponential, provided the resolutions are regular (Galil, Tseitin, cf. [Ga]).

References


