IBM Journal
of research and development

Vol. 25, No. 4, July 1981

Communication

Dexter Kozen

Positive First-Order Logic Is NP-Complete

ISSN 18-8646. Printed in U.S.A.

Dexter Kozen

Positive First-Order Logic Is NP-Complete

The decision problem for positive first-order logic with equality is NP-complete. More generally, if Σ is a finite set of atomic sentences (i.e., atomic formulas of the form $t_1 = t_2$ or $Rt_1 \cdots t_n$ containing no variables) and negations of atomic sentences and if ϕ is a positive first-order sentence, then the problem of determining whether ϕ is true in all models of Σ is NP-complete.

Introduction

Although it is undecidable in general whether a given sentence of first-order predicate logic is a theorem, algorithms have been discovered for various special cases; in some instances the line between the decidable and undecidable is finely drawn (see [1, 2]). Much effort has been directed toward the development of sophisticated techniques for the decidable special cases, such as resolution, paramodulation, the inverse method, and their various refinements (see [3]).

More recently, interest in the inherent complexity of computational problems has led to the successful complexity-theoretic taxonomy of several decision problems in logic (see, *e.g.*, [4–9]). In particular, Lewis [10] discusses the complexity of some solvable special cases of the decision problem for first-order logic.

In this note we show that the decision problem for positive first-order logic with equality is NP-complete [11]. More generally, if Σ is a finite set of atomic sentences (*i.e.*, atomic formulas of the form $t_1 = t_2$ or $Rt_1 \cdots t_n$ containing no variables) and negations of atomic sentences and if ϕ is a positive first-order sentence, then the problem of determining whether ϕ is true in all models of Σ (in symbols, $\Sigma \models \phi$) is NP-complete. Unlike the decidable special cases treated in [10], this problem places no restriction on the use of quantifiers.

The result of this note characterizes exactly the expressive power of positive sentences. Also, our nondeterministic polynomial-time algorithm is quite straightforward and does not use any of the sophisticated machinery of [12, 13], for example, yet it is optimal in the sense that its worst-case complexity cannot be significantly reduced unless P = NP.

The problem $\Sigma \models \phi$ for positive ϕ is of particular interest because many common combinatorial problems arising in computer science are special cases. For example, if G is a graph with nodes $c_1, \cdots, c_n$, Σ is the set

$$\{c_i \neq c_j \mid i \neq j\}$$

$$\cup \{c_i E c_j \mid (c_i, c_j) \text{ is an edge of } G\}$$

$$\cup \{\neg\, c_i E c_j \mid (c_i, c_j) \text{ is not an edge of } G\}$$

and ϕ is the positive sentence

$$\exists x_1 \cdots \exists x_n \left(\bigwedge_{1 \le i \le n} \bigvee_{1 \le j \le n} c_i = x_j\right) \wedge \left(\bigwedge_{1 \le i < n} x_i E x_{i+1}\right) \wedge x_n E x_1 ,$$

then $\Sigma \models \phi$ if and only if G has a Hamiltonian circuit.

The above few lines already show that the problem $\Sigma \models \phi$ is at least NP-hard, since the Hamiltonian circuit problem is known to be NP-complete (see [14]). We show below that the problem is NP-hard even for $\Sigma = \varnothing$.

Notation and terminology

The language L of first-order logic with equality consists of a countably infinite set $x_0, x_1, \cdots$ of *individual variables*, a countably infinite set $f, g, \cdots$ of *function symbols* for each arity $m \ge 0$, a countably infinite set $R, \cdots$ of *relation symbols* for each arity $m \ge 0$ (of which one is the binary *equality symbol* $=$), *logical symbols* $\wedge, \vee, \neg, \exists, \forall$,

and parentheses. Nullary function symbols are called *constants* and are denoted c, c_0, $\cdots$.

A *term* is a variable, a constant, or an expression $ft_1 \cdots t_m$, where f is an m-ary function symbol and $t_1, \cdots, t_m$ are terms. A term is *closed* if it contains no variables. Terms are denoted s, t, $\cdots$.

Formulas are defined inductively: If $t_1, \cdots, t_m$ are terms and R is an m-ary relation symbol, then $Rt_1 \cdots t_m$ is an *atomic formula*, and if ϕ, ψ are formulas, then $\phi \wedge \psi$, $\phi \vee \psi$, $\neg \phi$, $\exists x \phi$, and $\forall x \phi$ are. A formula is *closed* if it contains no free (unquantified) variables. A closed formula is called a *sentence*. A formula is *positive* if it contains no occurrence of $\neg$.

We write $\phi(x_1, \cdots, x_n)$ and $t(x_1, \cdots, x_n)$ to indicate that all free variables of formula ϕ and term t are among $x_1, \cdots, x_n$. If $\phi(x)$ is a formula and t is a term, then $\phi(t)$ denotes $\phi(x)$ with all free occurrences of x replaced by t.

A structure A for L consists of a set $|A|$ and an interpretation f^A, R^A for each function and relation symbol f, R. If f is an m-ary function symbol, then f^A is a function $|A|^m \rightarrow |A|$ (constants c^A are elements of $|A|$), and if R is an m-ary relation symbol, then R^A is a relation $|A|^m \rightarrow$ {**true, false**}. The equality symbol $=$ is always interpreted as the identity relation.

We write $A \models \phi$ if sentence ϕ is true in A. A sentence is *satisfiable* iff it is true in some structure and *valid* if it is true in all structures, *i.e.*, if its negation is not satisfiable. We write $\models \phi$ to denote that ϕ is valid.

Efficient representation of formulas as labeled graphs

Although formulas are officially strings of symbols, for the purpose of efficient computation we represent them as labeled rooted directed acyclic graphs. A constant c or variable x is represented as a single node labeled c or x, and a term $ft_1 \cdots t_m$ is represented as a rooted directed acyclic graph whose root node is labeled f and has m edges numbered $1, \cdots, m$ pointing to the representations of the terms $t_1, \cdots, t_m$.

The reason for this particular representation is that it allows consolidation of common subterms. For example, we often want to replace all free occurrences of a variable x in a term t with another term u, and the representation we have chosen allows us to do this by making some edges in the graph point to u instead of x. This avoids duplication of the term u so that the representation does not grow too big. Atomic formulas $s = t$ are represented as undirected edges between the roots of the representations of s and t.

The Herbrand structure

A particularly useful structure is the *Herbrand structure* T. The elements of T are the closed terms of L. Function symbols are given their syntactic interpretation

$$f^T(t_1, \cdots, t_m) = ft_1 \cdots t_m .$$

Relation symbols other than $=$ are interpreted as universally false.

The Herbrand structure is useful here because the validity problem for a positive sentence ϕ is equivalent to the problem of whether ϕ is true in T. This well-known fact is established in the following lemma.

• *Lemma 1*

For any positive sentence ϕ, $\models \phi$ iff $T \models \phi$.

Proof The direction ($\rightarrow$) is trivial. Now suppose ϕ is true in T, and let A be any finite or countable structure. We show that ϕ is true in A. It then follows from the Löwenheim-Skolem Theorem that ϕ is true in all structures.

Let $a_1, a_2, \cdots$ be a list of all the elements of A (repetitions allowed in case A is finite), and define $h(c_i) = a_i$. The function h extends uniquely to the set of all closed terms according to the rule

$$h(ft_1 \cdots t_m) = f^A(h(t_1), \cdots, h(t_m)).$$

Because $=^T$ is syntactic identity and all other R^T are universally false,

$$R^T(t_1, \cdots, t_m) \rightarrow R^A(h(t_1), \cdots, h(t_m)) \qquad (1)$$

for all terms $t_1, \cdots, t_m$ and relation symbols R (including $=$). It now follows inductively that for any positive $\phi(x_1, \cdots, x_n)$,

$$T \models \phi(t_1, \cdots, t_n) \rightarrow A \models \phi(h(t_1), \cdots, h(t_n)). \qquad (2)$$

The basis is exactly (1), and it is trivial to show that (2) is preserved by the logical operators $\vee$, $\wedge$, and $\exists$. The case for $\forall$ is almost as trivial but uses the fact that h is onto. □

Normal form

In order to show that the decision problem for a positive sentence ϕ is in NP, we first show that we can make certain assumptions about the form of ϕ without loss of generality. First we can assume that ϕ is in *prenex form*, that is, $\phi = Q_1 x_1 \cdots Q_n x_n \psi$ and ψ is quantifier free, since there is a simple polynomial-time algorithm for converting formulas to this form (see, *e.g.*, [15]). The quantifier string $Q_1 x_1 \cdots Q_n x_n$ is called the *prefix*, and the quantifier free part ψ is called the *matrix*. We can also assume that ϕ contains no relation symbols except $=$, since every

relation symbol other than = is universally false in T; thus any $Rt_1 \cdots t_m$ can be replaced by $c_0 = c_1$, which is also false in T.

We can also limit our attention to existential sentences, *i.e.*, those not containing any universal quantifiers. This is done by replacing each universally quantified variable y with $g(x_1, \cdots, x_k)$, where $x_1, \cdots, x_k$ are the variables which occur to the left of y in the quantifier prefix and g is a new function symbol. The resulting formula is valid just in case the original one was (see [1]).

NP-completeness of the validity problem for positive sentences

Let ϕ be a positive sentence. We show in this section that the problem of deciding whether $\models \phi$, or equivalently whether $T \models \phi$, is NP-complete. This is the special case $\Sigma = \varnothing$ of the more general validity problem $\Sigma \models \phi$, where Σ is a finite set of atomic sentences and negations of atomic sentences, which we treat in the next section.

By the arguments of the previous section, we can assume that ϕ is in prenex form and has only existential quantifiers. Since

$T \models \exists x_1 \cdots \exists x_n (\phi \vee \psi)$ iff either

$$T \models \exists x_1 \cdots \exists x_n \phi \text{ or } T \models \exists x_1 \cdots \exists x_n \psi, \tag{3}$$

we can reduce the problem *nondeterministically* to one in which the sentence has a conjunctive matrix, as follows. Suppose the matrix is of the form $\phi_1 \wedge \cdots \wedge \phi_k$ and each ϕ_i is of the form $\phi_{i,1} \vee \cdots \vee \phi_{i,m}$. To determine whether

$T \models \exists x_1 \cdots \exists x_n \phi_1 \wedge \cdots \wedge \phi_k$,

nondeterministically choose some ϕ_{i,j_i} from each ϕ_i and ask whether

$T \models \exists x_1 \cdots \exists x_n \phi_{1,j_1} \wedge \cdots \wedge \phi_{k,j_k}$.

If the original sentence is true in T, then the right choice will yield a sentence that is also true in T, by (3). If the original sentence is false in T, then no choice will yield a true sentence. This process can be repeated until all occurrences of $\vee$ have been eliminated.

Thus it remains to show how to test $T \models \exists x_1 \cdots \exists x_n \psi$, where ψ is conjunctive. This is an instance of the so-called *unification problem*, for which very efficient deterministic algorithms are known [16, 17]. We reproduce here the naive algorithm, which is slightly but not substantially less efficient.

• *Lemma 2*

There is a deterministic polynomial-time algorithm for deciding whether $T \models \phi$, where ϕ is a positive existential prenex conjunctive sentence.

Proof Let ϕ be the sentence $\exists x_1 \cdots \exists x_n \psi$, where ψ is a conjunction of atomic formulas $s = t$. The conjunction ψ is represented as a set of undirected edges on the graph representation of the terms in ψ.

If $s = t$ appears in ψ and s and t begin with different function symbols, then the sentence can immediately be declared false, because no choice of variables can satisfy $s = t$. If s and t are identical constants, then $s = t$ is trivially true and can be eliminated from ψ by removing the edge corresponding to $s = t$. If $s = fs_1 \cdots s_k$ and $t = ft_1 \cdots t_k$, then $s = t$ can be replaced with $s_1 = t_1 \wedge \cdots \wedge s_k = t_k$. This process is repeated until at least one side of each atomic formula in ψ is a variable. This takes only polynomial time and results in no significant increase in size, since we only manipulate edges in the graph.

We now show how to eliminate any variable x such that some $x = u$ appears in ψ. If x appears in u, then the entire formula can immediately be declared false, since no term can be equal to a proper subterm of itself. Otherwise, since all variables are existentially quantified, we can rearrange the prefix so that x is rightmost. Now ϕ is in the form

$$Q \exists x (x = u \wedge \sigma(x, \cdots)),$$

where Q is the part of the prefix not containing $\exists x$. Replace all occurrences of x in $\sigma(x, \cdots)$ with u to get $\sigma(u, \cdots)$. This is done by redirecting edges in the graph representation of the formula, so there is no substantial increase in size. Note that x does not occur in $\sigma(u, \cdots)$, since x does not occur in u. Now ϕ is of the form

$$Q \exists x (x = u \wedge \sigma(u, \cdots)),$$

which is equivalent to

$$Q ((\exists x\ x = u) \wedge \sigma(u, \cdots)),$$

since x does not occur in $\sigma(u, \cdots)$. But $\exists x\ x = u$ is trivially true in T for all choices of variables in Q, so we can delete it to get

$$Q \sigma(u, \cdots),$$

which does not contain the variable x. If x was the last variable in ϕ, then we declare ϕ true. The new formula is not substantially bigger than the original, due to our directed acyclic graph representation.

We repeat the entire procedure until it eventually halts, declaring ϕ either true or false. This must happen within polynomial time, since each pass eliminates a variable. □

• *Theorem 3*
The validity problem for positive sentences is NP-complete.

Proof The above sequence of lemmas shows that the problem is in NP. To show that it is NP-complete, we encode the Boolean satisfiability problem, a well-known NP-complete problem.

Let B be a Boolean formula with n variables $P_1, \cdots, P_n$. By DeMorgan's laws we can assume without loss of generality that all negations are applied to the P_i only. Let B' be formed by replacing each P_i by $x_i = c_1$ and each $\neg P_i$ by $x_i = c_0$. If B is satisfiable over {**true**, **false**}, then B' is satisfiable over $\{c_0, c_1\}$ by assigning c_1 to x_i if P_i is assigned **true** and c_0 to x_i if P_i is assigned **false**. Therefore, B' is satisfiable over T. Conversely, if B' is satisfiable over T, then B' is satisfiable over $\{c_0, c_1\}$, by reassigning any x_i not assigned to c_0 or c_1 to either c_0 or c_1. The monotonicity of B' guarantees that the new assignment also satisfies B'. From this we get a satisfying assignment for B in the obvious way. Thus B is satisfiable over {**true**, **false**} iff $T \models \exists x_1 \cdots \exists x_n B'$. □

The general problem

In this section we show that the more general positive validity problem $\Sigma \models \phi$ is NP-complete, where Σ is a finite set of atomic sentences or negations of atomic sentences. We established in the previous section that it is NP-hard, even in the special case $\Sigma = \varnothing$.

First we can assume without loss of generality that Σ contains only positive atomic sentences, since

$$\Sigma \cup \{\neg Rt_1 \cdots t_n\} \models \phi \text{ iff } \Sigma \models Rt_1 \cdots t_n \vee \phi .$$

Once we have eliminated negated atomic sentences from Σ, we can construct an Herbrand domain as above. This structure, denoted T/Σ, will consist of the closed terms T modulo the congruence relation $\approx$ induced by the equalities in Σ. That is, we define $\approx$ to be the smallest equivalence relation on elements of T such that $s \approx t$ for all identities $s = t$ in Σ, and whenever f is an m-ary function symbol and $s_i \approx t_i$, $1 \leq i \leq m$, then $fs_1 \cdots s_m \approx ft_1 \cdots t_m$. Equivalently, $s \approx t$ iff there is a sequence $s_0, s_1, \cdots, s_n$ with $s = s_0$ and $s_n = t$ such that s_{i+1} is obtained from s_i by replacing an occurrence of a subterm u by v, where $u = v \in \Sigma$. The elements of T/Σ are then the $\approx$-congruence classes $[t]$ of terms t. The function symbols are interpreted in T/Σ as

$$f^{T/\Sigma}([t_1], \cdots, [t_m]) = [ft_1 \cdots t_m] .$$

The relation symbol $=$ is interpreted as the identity relation; thus for closed terms s, t,

$$T/\Sigma \models s = t \text{ iff } s \approx t .$$

For the other relation symbols R, we define $R^{T/\Sigma}([t_1], \cdots, [t_m])$ = **true** if there exist $s_1, \cdots, s_m$ such that $s_i \approx t_i$ for $1 \leq i \leq m$ and $Rs_1 \cdots s_m \in \Sigma$; otherwise $R^{T/\Sigma}([t_1], \cdots, [t_m])$ = **false**.

The following lemma is a more general form of Lemma 1. The proof is similar and is left to the reader.

• *Lemma 4*
For any positive sentence ϕ and set of atomic sentences Σ, $\Sigma \models \phi$ iff $T/\Sigma \models \phi$. □

As before, we can assume ϕ is in prenex form and contains no universal quantifiers. Also, since (3) holds for T/Σ, we can nondeterministically make the matrix conjunctive. We now show that we can eliminate all relation symbols but $=$ from Σ and ϕ.

• *Lemma 5*
For any relation symbol R, let A_R be the set of all m-tuples of closed terms $t_1, \cdots, t_m$ such that $Rt_1 \cdots t_m$ appears in Σ. Then

$$T/\Sigma \models \forall x_1 \cdots \forall x_m \, (Rx_1 \cdots x_m \leftrightarrow \bigvee_{A_R} (x_1 = t_1 \wedge \cdots \wedge x_m = t_m)).$$

Proof This says that for all terms $s_1, \cdots, s_m$,

$$R^{T/\Sigma}([s_1], \cdots, [s_m]) \leftrightarrow \bigvee_{A_R} ([s_1] = [t_1] \wedge \cdots \wedge [s_m] = [t_m])$$
$$\leftrightarrow \bigvee_{A_R} (s_1 \approx t_1 \wedge \cdots \wedge s_m \approx t_m).$$

But this is exactly the definition of $R^{T/\Sigma}$. □

This result allows us to assume without loss of generality that neither Σ nor ϕ contains an occurrence of a relation symbol other than $=$, since each $Ru_1 \cdots u_m$ in ϕ can be replaced with $\bigvee_{A_R} \bigwedge_{1 \leq i \leq m} u_i = t_i$. Actually, $Ru_1 \cdots u_m$ will be replaced with one of the $\bigwedge_{1 \leq i \leq m} u_i = t_i$, where the $t_1, \cdots, t_m \in A_R$ is chosen nondeterministically. Once there are no more occurrences of R in ϕ, the presence or absence of any $Rt_1 \cdots t_m$ in Σ cannot affect the truth of ϕ, so we might as well remove all $Rt_1 \cdots t_m$ from Σ.

It will be more convenient for the remainder of the argument to work in T instead of in T/Σ. For this purpose let us append a symbol $\approx$ to L and interpret $\approx$ in T as the congruence relation on T induced by Σ, as defined above. Then

$$T/\Sigma \models s = t \text{ iff } T \models s \approx t,$$

and it follows by induction on formula structure that

$$T/\Sigma \models \phi([t_1], \cdots, [t_n]) \text{ iff } T \models \phi^{\sim}(t_1, \cdots, t_n)$$

for any formula $\phi(x_1, \cdots, x_n)$, where $\phi^{\approx}$ is the formula obtained from ϕ by replacing $=$ with $\approx$ throughout.

We now observe some properties of the relation $\approx$. If s and t are closed terms, define $s \sim t$ if s and t begin with the same m-ary function symbol and $s_i \approx t_i$ for all $1 \leq i \leq m$, where s_i and t_i are the ith maximal proper subterms of s and t, respectively. In case s and t are constants, $s \sim t$ iff $s = t$. Let S be the set of all subterms of terms in Σ.

• *Lemma 6*
For any closed terms s, t, $s \approx t$ iff either $s \sim t$ or there exist $u, v \in S$ such that $s \sim u \approx v \sim t$.

Proof Recall that $s \approx t$ iff there exists a sequence $s = s_0, s_1, \cdots, s_n = t$ such that s_{i+1} is obtained from s_i by substituting v for some occurrence of u, where $u \approx v \in \Sigma$. If none of these substitutions is ever made at the root of some s_i, then $s \sim t$. Otherwise let s_i, s_{i+1} be the first time a substitution is made at the root, and let s_j, s_{j+1} be the last time. Take $u = s_i$ and $v = s_{j+1}$. □

• *Lemma 7*
Let $u(x)$ be a term with one occurrence of the variable x, but $u \neq x$. If $t \approx u(t)$, then $t \approx s$ for some $s \in S$.

Proof If $t \approx u(t)$, then

$$t \approx u(t) \approx u(u(t)) \approx u(u(u(t))) \approx \cdots \approx u^k(t) \approx \cdots .$$

Let k be greater than the depth of t and let π be the path from the root of $u^k(x)$ to x. Then π is of length at least k, and t occurs in $u^k(t)$ at position π. If $u^k(t) \approx t$ via the sequence $u^k(t) = s_0, s_1, \cdots, s_n = t$, then eventually a substitution must be applied for the first time to an ancestor of the subterm at π, for if not then the depth of the terms in the sequence would never fall below k, and t could never be derived. At that point, the subterm at position π must be in S, since it appears in some $u \approx v \in \Sigma$, and it must be congruent to t, since previously the only substitutions that have been made have been to subterms of the term at π or to occurrences of subterms disjoint from the term at π. □

• *Lemma 8*
Let $\psi(x_1, \cdots, x_n)$ be a conjunction of atomic formulas of the form $s \approx t$. There is a nondeterministic polynomial-time algorithm to determine whether $T \models \exists x_1 \cdots \exists x_n \psi$.

Proof The proof is very similar to that of Lemma 2 with some notable exceptions, so we outline the argument with emphasis on the novel points.

First we reduce the problem to one in which each atomic formula in ψ is of the form either $x \approx u$, where x is a variable, or $s \approx t$, where s and t are closed terms. (This was done deterministically above, but here it requires nondeterministic choices.) If $u \approx v \in \psi$ and neither u nor v are variables but one of u, v contains a variable, then $u \approx v$ is equivalent to

$$u \sim v \vee \bigvee (u \sim s \wedge s \approx t \wedge t \sim v) ,$$

where the join is taken over all $s, t \in S$, by Lemma 6. We can replace $u \approx v$ with either $u \sim v$ or $u \sim s \wedge s \approx t \wedge t \sim v$ for some $s, t \in S$, where the choice is made nondeterministically. Now for each $r \sim w$ produced in this way, neither r nor w is a variable. If r and w begin with different function symbols, then the sentence can immediately be declared false, by definition of $\sim$. Otherwise $r \sim w$ can be replaced with $r_1 \approx w_1 \wedge \cdots \wedge r_m \approx w_m$, where r_i and w_i are the ith maximal proper subterms of r and w, respectively; or if r and w are both constants, then $r = w$, so $r \sim w$ can be deleted from the conjunction. This process is repeated until every atomic formula in ψ is of the form $x \approx u$ or $s \approx t$, where $s, t \in S$.

For every $s \approx t \in \psi$ such that s and t are both closed terms, it is decidable in polynomial time whether $s \approx t$, using the polynomial-time congruence closure algorithm of [18]. If not $s \approx t$, then the entire sentence can immediately be declared false, and otherwise $s \approx t$ can be eliminated from the conjunction ψ. Also, any $x \approx x$ can be eliminated from ψ. We repeat this until every formula in ψ is of the form $x \approx u$, where x is a variable and $u \neq x$.

Now we eliminate some variable x such that $x \approx u$ appears in ψ. If x does not appear in u, then x can be eliminated from ψ exactly as in the previous section. However, unlike the previous section, if x appears in u, then the sentence is not necessarily false. But if this is the case, $x \approx u$ implies $\bigvee_{t \in S} x \approx t$ by Lemma 7; therefore, $t \in S$ can be chosen nondeterministically and $x \approx t$ appended to ψ. Then t can be used to eliminate the variable x, since t contains no variables.

The above process is repeated until it halts and declares the formula either true or false. This must occur in polynomial time, since each pass eliminates a variable. □

Combining Lemmas 4, 5, and 8, we have proved

• *Theorem 9*
The problem $\Sigma \models \phi$ is NP-complete. □

Although the unification problem (Lemma 2) has an efficient deterministic algorithm, the algorithm given for its generalized version (Lemma 8) is nondeterministic. It

is interesting to note that the problem of Lemma 8 is itself NP-complete; therefore Lemma 2 does not hold in the general case unless P = NP.

Acknowledgments

I sincerely thank Jan Mycielski, Richard Statman, and an anonymous referee, whose ideas have greatly improved the paper. Mycielski and the referee observed that the construction leading to Lemma 2 reduces the problem to an instance of unification. The observation following Theorem 9 is due to the referee. Statman provided the reduction of Lemma 5.

References and note

1. B. Dreben and W. B. Goldfarb, *The Decision Problem: Solvable Classes of Quantificational Formulas*, Addison-Wesley Publishing Company, Reading, MA, 1979.
2. H. R. Lewis, *Unsolvable Classes of Quantificational Formulas*, Addison-Wesley Publishing Company, Reading, MA, 1979.
3. *Machine Intelligence*, Vol. 4, B. Meltzer and D. Michie, Eds., American Elsevier, New York, 1969.
4. J. Ferrante and C. Rackoff, "The Computational Complexity of Logical Theories," *Springer-Verlag Lecture Notes in Mathematics 718*, 1979.
5. M. J. Fischer and M. O. Rabin, "Super-exponential Complexity of Presburger Arithmetic," *SIAM-AMS Proceedings VII*, American Mathematical Society, Providence, RI, 1974.
6. L. Berman, "The Complexity of Logical Theories," *Theor. Comput. Sci.* **10,** 71–77 (1980).
7. D. Kozen, "Complexity of Boolean Algebras," *Theor. Comput. Sci.* **10,** 221–247 (1980).
8. D. Oppen, "An Upper Bound on the Complexity of Presburger Arithmetic," *J. Comput. Syst. Sci.* **16,** 323–332 (1978).
9. C. Rackoff, "On the Complexity of the Theories of Weak Direct Products: Preliminary Report," *Proceedings of the 6th ACM Symposium on Theory of Computing*, May 1974, pp. 149–160.
10. H. R. Lewis, "Complexity of Solvable Cases of the Decision Problem for the Predicate Calculus," *Proceedings of the 19th IEEE Symposium on Foundations of Computer Science*, Oct. 1978, pp. 35–47.
11. Part of this research was done at Cornell University and supported by NSF grant DCR75-09433.
12. G. Robinson and L. Wos, "Paramodulation and Theorem-Proving in First-Order Theories with Equality," *Machine Intelligence*, Vol. 4, B. Meltzer and D. Michie, Eds., American Elsevier, New York, 1969.
13. E. E. Sibert, "A Machine-Oriented Logic Incorporating the Equality Relation," *Machine Intelligence*, Vol. 4, B. Meltzer and D. Michie, Eds., American Elsevier, New York, 1969.
14. M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*, Freeman, San Francisco, 1979.
15. S. C. Kleene, *Introduction to Metamathematics*, Vol. I, 7th Ed., North-Holland Publishing Company, Amsterdam, 1974.
16. J. A. Robinson, "Computational Logic: The Unification Computation," *Machine Intelligence*, Vol. 6, B. Meltzer and D. Michie, Eds., American Elsevier, New York, 1971, pp. 63–72.
17. M. S. Paterson and M. N. Wegman, "Linear Unification," *J. Comput. Syst. Sci.* **16,** 158–167 (1978).
18. D. Kozen, "Complexity of Finitely Presented Algebras," *Proceedings of the 9th ACM Symposium on Theory of Computing*, May 1977, pp. 164–177.

Received January 8, 1981; revised February 9, 1981

The author is located at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598.