Eliminating definitions and Skolem functions in first-order logic

JEREMY AVIGAD
Carnegie Mellon University

From proofs in any classical first-order theory that proves the existence of at least two elements, one can eliminate definitions in polynomial time. From proofs in any classical first-order theory strong enough to code finite functions, including sequential theories, one can also eliminate Skolem functions in polynomial time.

Categories and Subject Descriptors: F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic—*Proof Theory*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: definitions, Skolem functions, proof complexity, lengths of proofs

1. INTRODUCTION

When working with a first-order theory, it is often convenient to use definitions. That is, if $\varphi(\vec{x})$ is a first-order formula with the free variables shown, one can introduce a new relation symbol R to abbreviate φ, with defining axiom $\forall \vec{x}\ (R(\vec{x}) \leftrightarrow \varphi(\vec{x}))$. Of course, this definition can later be eliminated from a proof, simply by replacing every instance of R by φ. But suppose the proof involves nested definitions, with a sequence of relation symbols $R_0, \ldots, R_k$ abbreviating formulae $\varphi_0, \ldots, \varphi_k$, where each φ_i may have multiple occurrences of $R_0, \ldots, R_{i-1}$. In that case, the naive elimination procedure described above can yield an exponential increase in the length of the proof.

In Section 2, I show that if the underlying theory proves that there are at least two elements in the universe, a more careful translation allows one to eliminate the new definitions with at most a polynomial increase in length. In fact, I will describe an explicit algorithm that can be implemented in polynomial time. The proof is not difficult, but it relies on the assumption that equality is included in the logic. A similar trick has been used by Solovay in simulating iterated definitions efficiently, as discussed in Section 3.2 of [Pudlák 1998]. Consequently, the result proved here may be folklore, but to my knowledge it has not appeared in the literature, and it is needed in Section 3.

It is also sometimes convenient, in a first-order setting, to introduce Skolem func-

This work has been partially supported by NSF grant DMS-070600. This is a revised, corrected, and slightly expanded version of [Avigad 2001b].

tions. If $\varphi(\vec{x}, y)$ is any formula with the free variables shown and f is a new function symbol, one can add an axiom, $\forall \vec{x}, y\ (\varphi(\vec{x}, y) \rightarrow \varphi(\vec{x}, f(\vec{x})))$, asserting, in words, "if any y satisfies $\varphi(\vec{x}, y)$, $f(\vec{x})$ does." There is an easy model-theoretic proof of the fact that this does not alter the set of consequences in the original language: any first-order model of the original theory can be expanded to a model where f denotes such a choice function. Explicit syntactic proofs of this fact are, however, somewhat more difficult. The first such proof appears in Hilbert and Bernays' *Grundlagen der Mathematik* [1939], using the epsilon substitution method; a proof by Maehara using cut-elimination is discussed in [Takeuti 1987]; and another proof due to Shoenfield is found in [Shoenfield 2001] (see also the discussion in [Schwichtenberg 1979]). All these procedures are, unfortunately, worse than exponential.

In Section 3, I show that if the underlying theory allows for a modicum of coding, one can also eliminate Skolem functions in polynomial time. The idea is to use an internal, iterated forcing argument to add the new functions. The forcing conditions involved are finite approximations to the Skolem functions being added, so the constraint on the underlying theory is that it provides an adequate representation of finite functions. The specific requirements are spelled out below; any sequential theory of arithmetic meets these criteria. While forcing methods have been used to establish lower bounds in proof complexity (see [Ajtai 1988; Krajíček 1995; Paris and Wilkie 1985]), here they are used to establish upper bounds; similar forcing arguments can be found in [Avigad 1996; 2000; 2001a; 2001c].

The question as to whether or not definitions can be eliminated efficiently from propositional proof systems is a major open question in the field of proof complexity. The results here show that the answer is "yes" for most *first-order* proof systems, though the most general statement of the problem is equivalent to the propositional version. Issues related to Skolem functions are similarly important to computer science, since most automated search procedures use Skolemization in one form or another.

The question as to the increase in proof length when eliminating a single Skolem function from a proof in pure first-order logic is listed as open problem 22 in [Clote and Krajíček 1993]. Once again, though these results here do not settle the most general statement of the problem, they show that for many natural theories such an efficient elimination is possible. So, at least in principle, it does not hurt to use Skolem functions when searching for proofs, even if the ultimate goal is to have a proof in ordinary first-order logic. In Section 4, I discuss some questions that remain.

2. ELIMINATING DEFINITIONS

If d is a proof of a sentence ψ from a set of axioms Γ in first-order logic, then $|d|$ denotes the length of d, according to the number of symbols. Good general references on the lengths of proofs are [Krajíček 1995] and [Pudlák 1998].

In this section and the next I will show that in certain circumstances one can eliminate definitions and/or Skolem functions from a proof d in such a way that the length of the resulting proof is bounded by a polynomial in $|d|$. In doing so, I will not make an effort to compute the exact polynomial; rather, I will repeatedly appeal to the fact that the set of polynomials in $|d|$ is closed under addition, multi-

plication, and composition. Indeed, it will be clear from the proofs that in fact all the translations considered can be carried out in polynomial time.

By "first-order logic," I mean first-order logic with equality, in any of the standard natural deduction calculi, Hilbert-style calculi, or sequent calculi with cut described in [Troelstra and Schwichtenberg 2000]. By a theorem due to Krajíček, up to polynomial-time equivalence it does not matter whether we take proofs to be given by trees or sequences of lines (see Section 4 of [Pudlák 1998], or Section 4.5 of [Krajíček 1995] for the propositional case). In fact, the proof of Theorem 2.2 only assumes that there is a representation of $\varphi \to \psi$ which uses φ only once. If $\leftrightarrow$ is assumed to be one of the basic connectives, one can simplify the central argument somewhat; but the proof below works in either case.

I will use the following conventions: $\vec{x}$ and $\vec{t}$ denote sequences of variables and terms, respectively, and typically their lengths can be inferred from the context. Introducing a formula as $\varphi(\vec{x})$ only serves to distinguish the sequence of variables $\vec{x}$, after which $\varphi(\vec{t})$ denotes the result of simultaneously substituting $\vec{t}$ for $\vec{x}$, renaming bound variables in φ if necessary.

Definition 2.1. Let Γ be a set of first-order sentences in a language L. Say that Γ has an *efficient elimination of definitions* if there is a polynomial-time algorithm that behaves as follows: whenever $R_0(\vec{x}_0), \ldots, R_k(\vec{x}_k)$ are new relation symbols of various arities, $\varphi_0(\vec{x}_0), \ldots, \varphi_k(\vec{x}_k)$ are formulae such that each φ_i is in the language $L \cup \{R_0, \ldots, R_{i-1}\}$, and d is a proof of a formula ψ in L from

$$\Gamma \cup \{\forall \vec{x}_0\ (R_0(\vec{x}_0) \leftrightarrow \varphi_0(\vec{x}_0)), \ldots, \forall \vec{x}_k\ (R_k(\vec{x}_k) \leftrightarrow \varphi_k(\vec{x}_k))\},$$

then, on input d, the algorithm returns a proof d' of ψ from Γ using only formulae in L.

Note that, in particular, the definition implies that there is a polynomial p such that each proof d involving definitions is mapped to a proof d' without them, with $|d'| \leq p(|d|)$. This definition is monotone in Γ: if Γ has an efficient elimination of definitions and $\Gamma' \supseteq \Gamma$ then, by the deduction theorem, Γ' has an efficient elimination of definitions as well. The main theorem in this section is the following:

THEOREM 2.2. *$\{\exists x, y\ (x \neq y)\}$ has an efficient elimination of definitions.*

PROOF. The proof will occupy most of this section. Let $R_0, \ldots, R_k, \varphi_0, \ldots, \varphi_k, \psi$, and d be as in the definition. We can assume that each of the defining axioms occurs at least once in the proof, since if the axiom for R_i does not occur in the proof we can replace each occurrence of R_i by an arbitrary sentence, say $\forall x\ (x = x)$. As a result, we can assume that k and $|\varphi_0|, \ldots, |\varphi_k|$ are all less than $|d|$, and so it suffices to bound the length of the final proof by a polynomial in these values.

Let a and b be new constant symbols. It suffices to find an efficient proof of ψ from $\{a \neq b\}$, since, given a proof of $a \neq b \to \psi$, we can replace a and b by variables to obtain a proof of ψ from $\exists x, y\ (x \neq y)$.

Put simply, the idea is to use a and b as truth values, and to use first-order quantifiers to avoid repeating definitions. It will help to consider a simple example first.

Example. Suppose φ_{i+1} is the formula $R_i(\vec{s}) \wedge \neg R_i(\vec{t})$. We can express φ_{i+1} as an equivalent formula in which R_i occurs only once, in the following way. Let $\theta(v, v')$

be the formula

$$\forall \vec{x}, y\ ((R_i(\vec{x}) \leftrightarrow y = a) \rightarrow (\vec{x} = \vec{s} \rightarrow y = v) \wedge (\vec{x} = \vec{t} \rightarrow y = v')).$$

Then φ_{i+1} is equivalent to

$$\forall v, v'\ (\theta(v, v') \rightarrow (v = a \wedge v' \neq a)).$$

This example contains the essence of the entire proof; in particular, θ plays the role of the formula *Eval* below. In the general case, we have a few additional concerns:

(1) there may be quantifiers in the various φ_i;
(2) all of the symbols $R_0, \ldots, R_{i-1}$ may occur in φ_i; and
(3) $\leftrightarrow$ may not be a symbol in the language (using implication twice in the example above would require another instance of R_i).

With respect to item 1, it will help to assume that all the definitions are given by prenex formulae, and we can do so without loss of generality. If the propositional connectives are among $\{\wedge, \vee, \rightarrow, \neg\}$, this is so because any formula involving these connectives can be proved equivalent to one that is prenex, with a proof whose length is bounded by a polynomial in the length of the original formula. On the other hand, if, say, $\leftrightarrow$ is a propositional connective, one can introduce additional definitions to abbreviate subformulae and ensure that all the definitions are prenex. Alternatively, one can first use definitions to eliminate $\leftrightarrow$ as in the proof of Corollary 2.5, and then proceed as before.

Henceforth, if θ is a formula with a relation symbol $R(\vec{y})$ and $\eta(\vec{y})$ is a formula with the free variables shown, it will be convenient to write $\theta[\eta/R]$ for the result of replacing each atomic formula $R(\vec{t})$ by $\eta(\vec{t})$. At other times, I will write $\theta[R(t_1, \ldots, t_m)]$ to indicate that an atomic formula $R(t_1, \ldots, t_m)$ occurs in the quantifier-free formula θ; thereafter $\theta[\eta]$ denotes the result of replacing $R(t_1, \ldots, t_m)$ by η. While this notation is potentially problematic, the intention should always be clear from the context.

For notational convenience, we may assume that all of the relations R_i have the same arity. To address item 2 above, we will need a way of representing the numbers $0, \ldots, k$. To that end, let $z_0, \ldots, z_k$ be a sequence of variables, write $\overline{0}$ for the sequence $a, b, b, b, \ldots$, $\overline{1}$ for the sequence $b, a, b, b, \ldots$, and, more generally, $\overline{j}$ for the sequence of length $k+1$ that has an a in position j and b's elsewhere.

Finally, to address item 3, we will represent both positive and negative instances of each definition.

Our strategy will be to define a sequence of formulae $\hat{\varphi}_0(\vec{z}, u, \vec{x}), \ldots \hat{\varphi}_k(\vec{z}, u, \vec{x})$, with length bounded by a polynomial in $|d|$, such that for each $i \leq k$ the following equivalences are all provable from $a \neq b$:

—$\hat{\varphi}_i(\overline{j}, a, \vec{x}) \leftrightarrow \hat{\varphi}_{i-1}(\overline{j}, a, \vec{x})$, for each $j < i$
—$\hat{\varphi}_i(\overline{j}, b, \vec{x}) \leftrightarrow \neg\hat{\varphi}_{i-1}(\overline{j}, a, \vec{x})$, for each $j < i$
—$\forall \vec{x}\ (\hat{\varphi}_i(\overline{i}, a, \vec{x}) \leftrightarrow \varphi_i(\vec{x})[\hat{\varphi}_{i-1}(\overline{0}, a, \vec{x})/R_0, \ldots, \hat{\varphi}_{i-1}(\overline{i-1}, a, \vec{x})/R_{i-1}])$
—$\forall \vec{x}\ (\hat{\varphi}_i(\overline{i}, b, \vec{x}) \leftrightarrow \neg\varphi_i(\vec{x})[\hat{\varphi}_{i-1}(\overline{0}, a, \vec{x})/R_0, \ldots, \hat{\varphi}_{i-1}(\overline{i-1}, a, \vec{x})/R_{i-1}])$.

In other words, for each i and $j \leq i$, $\hat{\varphi}_i(\bar{j}, a, \vec{x})$ is an efficient representation of R_j, and $\hat{\varphi}_i(\bar{j}, b, \vec{x})$ is an efficient representation of $\neg R_j$. As noted above, we will use quantifiers and equality so that only a single instance of $\hat{\varphi}_i$ is used in the definition of $\hat{\varphi}_{i+1}$. Note that the clauses above imply that for each i and $j \leq i$, we have $\hat{\varphi}_i(\bar{j}, a, \vec{x}) \leftrightarrow \neg\hat{\varphi}_i(\bar{j}, b, \vec{x})$.

The construction. The sequence $\hat{\varphi}_0, \ldots, \hat{\varphi}_k$ is defined recursively. Start by taking $\hat{\varphi}_0(\vec{z}, u, \vec{x})$ to be the formula

$$(u = a \rightarrow \varphi_0(\vec{x})) \wedge (u = b \rightarrow \neg\varphi_0(\vec{x})).$$

For $i > 0$, assuming $\hat{\varphi}_0, \ldots, \hat{\varphi}_{i-1}$ have been defined, the following shows how to determine $\hat{\varphi}_i$. Since we are assuming that all the definitions are prenex, $\varphi_i(\vec{x})$ is of the form

$$Q_1 y_1 \ldots Q_m y_m \; \tilde{\varphi}[R_0(\vec{t}_{0,0}), \ldots, R_0(\vec{t}_{0,l_0}), \ldots, \quad R_{i-1}(\vec{t}_{i-1,0}), \ldots, R_{i-1}(\vec{t}_{i-1,l_{i-1}})],$$

where $\tilde{\varphi}$ is quantifier-free and the sequence in square brackets shows all instances of atomic formulae in $\tilde{\varphi}$ involving $R_0, \ldots, R_{i-1}$. In general, the sequences of terms $\vec{t}_{j,p}$ depend on the quantified variables $y_1, \ldots, y_m$ as well as the free variables $\vec{x}$ of φ_i, but I will not display these variables explicitly. Our task is to write down a formula $\hat{\varphi}_i(\vec{z}, u, \vec{x})$ such that

(1) for each $j < i$, $\hat{\varphi}_i(\bar{j}, a, \vec{x})$ is equivalent to $\hat{\varphi}_{i-1}(\bar{j}, a, \vec{x})$;

(2) for each $j < i$, $\hat{\varphi}_i(\bar{j}, b, \vec{x})$ is equivalent to $\neg\hat{\varphi}_{i-1}(\bar{j}, a, \vec{x})$;

(3) $\hat{\varphi}_i(\bar{i}, a, \vec{x})$ is equivalent to the displayed formula above, with each $R_j(\vec{t}_{j,p})$ replaced by $\hat{\varphi}_{i-1}(\bar{j}, a, \vec{t}_{j,p})$;

(4) $\hat{\varphi}_i(\bar{i}, b, \vec{x})$ is equivalent to the negation of the formula just described; and

(5) in the definition of $\hat{\varphi}_i$, $\hat{\varphi}_{i-1}$ is used only once.

Items 1–4 are just a restatement of the desiderata indicated above; 5 will ensure that the $\hat{\varphi}_i$ can be constructed in polynomial time.

In order to do 3 and 4 simultaneously, we need duplicate copies of some of the variables and terms. Let $Q'_1, \ldots, Q'_m$ denote the quantifiers dual to $Q_1, \ldots, Q_m$. Pick a new sequence of variables $y'_1, \ldots, y'_m$, and let

$$\vec{t}'_{0,0}, \ldots, \vec{t}'_{0,l_0}, \ldots, \vec{t}'_{i-1,0}, \ldots, \vec{t}'_{i-1,l_{i-1}}$$

denote the sequences of terms obtained by replacing the $y_1, \ldots, y_m$ by $y'_1, \ldots, y'_m$ in each $\vec{t}_{j,p}$. Finally, let

$$\begin{aligned} &v_{0,0}, \ldots, v_{0,l_0}, \ldots, v_{i-1,0}, \ldots, v_{i-1,l_{i-1}} \\ &v'_{0,0}, \ldots, v'_{0,l_0}, \ldots, v'_{i-1,0}, \ldots, v'_{i-1,l_{i-1}} \\ &v''_0, \ldots, v''_{i-1} \end{aligned}$$

be sequences of new variables. We will use the variables $v_{j,p}$ to represent the truth values of $\hat{\varphi}_{i-1}(\bar{j}, a, \vec{t}_{j,p})$, the variables $v'_{j,p}$ to represent the truth values of $\hat{\varphi}_{i-1}(\bar{j}, a, \vec{t}'_{j,p})$, and the variables v''_j to represent the truth values of $\hat{\varphi}_{i-1}(\bar{j}, a, \vec{x})$, where the "truth value" is a if the corresponding formula is true, and b if it is false.

The formula $\hat{\varphi}_i(\vec{z}, u, \vec{x})$ is defined to be

$$Q_1 y_1 \ldots Q_m y_m \; Q'_1 y'_1 \ldots Q'_m y'_m \; \forall \vec{v}, \vec{v}', \vec{v}'' \; \Big(Eval(\vec{v}, \vec{v}', \vec{v}'') \rightarrow$$
$$\bigwedge_{j<i} (\vec{z} = \overline{j} \wedge u = a \rightarrow v''_j = a) \wedge \bigwedge_{j<i} (\vec{z} = \overline{j} \wedge u = b \rightarrow v''_j \neq a) \wedge$$
$$(\vec{z} = \overline{i} \wedge u = a \rightarrow \tilde{\varphi}[v_{0,0} = a, \ldots, v_{0,l_0} = a, \ldots, v_{i-1,0} = a, \ldots, v_{i-1,l_{i-1}} = a]) \wedge$$
$$(\vec{z} = \overline{i} \wedge u = b \rightarrow \neg\tilde{\varphi}[v'_{0,0} = a, \ldots, v'_{0,l_0} = a, \ldots, v'_{i-1,0} = a, \ldots, v'_{i-1,l_{i-1}} = a]) \Big)$$

where $Eval(\vec{v}, \vec{v}', \vec{v}'')$ is the formula

$$\forall \vec{r} \, \forall s \in \{a, b\} \; \forall \vec{w} \; \Big(\hat{\varphi}_{i-1}(\vec{r}, s, \vec{w}) \rightarrow \bigwedge_{j<i} (\vec{r} = \overline{j} \wedge \vec{w} = \vec{x} \rightarrow v''_j = s) \wedge$$
$$\bigwedge_{j<i} \bigwedge_{p \leq l_j} (\vec{r} = \overline{j} \wedge \vec{w} = \vec{t}_{j,p} \rightarrow v_{j,p} = s) \wedge$$
$$\bigwedge_{j<i} \bigwedge_{p \leq l_j} (\vec{r} = \overline{j} \wedge \vec{w} = \vec{t}_{j,p} \rightarrow v'_{j,p} = s) \Big).$$

Here $\forall s \in \{a, b\} \; \theta$ abbreviates $\forall s \; (s = a \vee s = b \rightarrow \theta)$. Note that $Eval(\vec{v}, \vec{v}', \vec{v}'')$ also depends on the free variables $\vec{x}, \vec{y}, \vec{y}'$ (because the terms $t_{j,p}$ and $t'_{j,p}$ do), but I will continue to leave these variables implicit.

First, let us check that each $\hat{\varphi}_i(\vec{x}_i, u)$ satisfies the right equivalences, and then let us worry about the length. Inductively we know, for each $j \leq i - 1$, that

$$\forall \vec{x} \; (\hat{\varphi}_{i-1}(\overline{j}, a, \vec{x}) \leftrightarrow \neg\hat{\varphi}_{i-1}(\overline{j}, b, \vec{x}))$$

is provable from $a \neq b$. We can use this to show

$$\forall \vec{x}, \vec{y}, \vec{y}' \; \exists \vec{v}, \vec{v}', \vec{v}'' \; Eval(\vec{v}, \vec{v}', \vec{v}'')$$

as well as

$$\forall \vec{x}, \vec{y}, \vec{y}', \vec{v}, \vec{v}', \vec{v}'' \; \Big(Eval(\vec{v}, \vec{v}', \vec{v}'') \rightarrow \bigwedge_{j<i} (v''_j = a \leftrightarrow \hat{\varphi}_{i-1}(\overline{j}, a, \vec{x})) \wedge$$
$$\bigwedge_{j<i} \bigwedge_{p<l_j} (v_{j,p} = a \leftrightarrow \hat{\varphi}_{i-1}(\overline{j}, a, \vec{t}_{j,p})) \wedge$$
$$\bigwedge_{j<i} \bigwedge_{p<l_j} (v'_{j,p} = a \leftrightarrow \hat{\varphi}_{i-1}(\overline{j}, a, \vec{t}_{j,p})) \Big).$$

But then, going back to the definition of $\hat{\varphi}_i$, we see that for $j < i$, $\hat{\varphi}_i(\overline{j}, a, \vec{x})$ is equivalent to $\hat{\varphi}_{i-1}(\overline{j}, a, \vec{x})$, and $\hat{\varphi}_i(\overline{j}, b, \vec{x})$ is equivalent to $\neg\hat{\varphi}_{i-1}(\overline{j}, a, \vec{x})$. Also, $\hat{\varphi}_i(\overline{i}, a, \vec{x})$ is equivalent to

$$Q_1 y_1 \ldots Q_m y_m \tilde{\varphi}[\hat{\varphi}_{i-1}(\overline{0}, a, \vec{t}_{0,0}), \ldots, \hat{\varphi}_{i-1}(\overline{0}, a, \vec{t}_{0,l_0}), \ldots,$$
$$\hat{\varphi}_{i-1}(\overline{i-1}, a, \vec{t}_{i-1,0}), \ldots, \hat{\varphi}_{i-1}(\overline{i-1}, a, \vec{t}_{i-1,l_{i-1}})]$$

and so we have

$$\hat{\varphi}_i(\overline{i}, a, \vec{x}) \leftrightarrow \varphi_i(\vec{x})[\hat{\varphi}_{i-1}(\overline{0}, a, \vec{x})/R_0, \ldots, \hat{\varphi}_{i-1}(\overline{i-1}, a, \vec{x})/R_{i-1}];$$

and $\hat{\varphi}_i(\bar{i}, b, \vec{x})$ is equivalent to

$$Q'_1 y'_1 \dots Q'_m y'_m \neg\tilde{\varphi}[\hat{\varphi}_{i-1}(\overline{0}, a, \vec{t}_{0,0}), \dots, \hat{\varphi}_{i-1}(\overline{0}, a, \vec{t}_{0,l_0}), \dots, \hat{\varphi}_{i-1}(\overline{i-1}, a, \vec{t}_{i-1,0}), \dots, \hat{\varphi}_{i-1}(\overline{i-1}, a, \vec{t}_{i-1,l_{i-1}})]$$

and so we have

$$\hat{\varphi}_i(\bar{i}, b, \vec{x}) \leftrightarrow \neg\varphi_i(\vec{x})[\hat{\varphi}_{i-1}(\overline{0}, a, \vec{x})/R_0, \dots, \hat{\varphi}_{i-1}(\overline{i-1}, a, \vec{x})/R_{i-1}],$$

as required.

As far as length is concerned, it is not hard to check that the number of symbols occurring in $\hat{\varphi}_i$ apart from the instance of $\hat{\varphi}_{i-1}$ can be bounded by a polynomial in $|d|$ (in fact, even a linear one). In other words, there is a polynomial p such that for each i we have $|\hat{\varphi}_i| \leq p(|d|) + |\hat{\varphi}_{i-1}|$, and hence $|\hat{\varphi}_i| \leq (i+1)p(|d|) \leq |d|p(|d|)$. It is not hard to see, moreover, that the $\hat{\varphi}_i$ can be constructed in time polynomial in $|d|$. Similarly, one can efficiently construct proofs of the necessary equivalences, and there are only polynomially many of them. Translating the original proof is now straightforward, using $\hat{\varphi}_k(\bar{i}, a, \vec{x})$ in place of $R_i(\vec{x})$ for each i.

This completes the proof of Theorem 2.2. □

We have handled the case where there are at least two elements in the universe. On the other hand, on the assumption that there is only one element of the universe, we are reduced to propositional logic.

PROPOSITION 2.3. *$\{\forall x, y\ (x = y)\}$ has efficient elimination of definitions if and only if the corresponding assertion holds for propositional logic.*

PROOF. Assuming $\forall x, y\ (x = y)$, every atomic formula $R(t_1, \dots, t_k)$ is equivalent to $R(c, \dots, c)$, where c is the only element of the universe; $t_1 = t_2$ is always true; and quantifiers have no effect. To be more precise, let "the propositional simplification of ψ" denote the result of deleting all the quantifiers in ψ, replacing all atomic formulae $R(t_1, \dots, t_k)$ by a propositional variable R, and replacing $t_1 = t_2$ by a fixed tautology. Then any first-order proof of $\forall x, y\ (x = y) \to \psi$ can be translated efficiently to a propositional proof of the propositional simplification of ψ, and vice-versa. □

This implies that the general problem of eliminating definitions from proofs in pure first-order logic is as hard (and as easy) as the propositional case.

THEOREM 2.4. *$\emptyset$ has an efficient elimination of definitions if and only if the corresponding assertion holds for propositional logic.*

PROOF. It is a straightforward exercise to check that $\{\varphi \vee \psi\}$ has an efficient elimination of definitions if and only if $\{\varphi\}$ and $\{\psi\}$ both do. In particular, $\emptyset$ has an efficient elimination of definitions if and only if $\{\forall x, y\ (x = y)\}$ and $\{\exists x, y\ (x \neq y)\}$ do. □

The question as to whether one can eliminate definitions from propositional logic efficiently (or even with a polynomial bound on the length of proof) is a major open problem in proof complexity; see [Krajíček 1995; Pudlák 1998].

As a corollary of Theorem 2.2, we have that one can eliminate $\leftrightarrow$ from standard proof systems in polynomial time. For propositional proof systems the proof (due to Reckhow, using a method by Spira; see [Krajíček 1995]) is considerably more difficult.

COROLLARY 2.5. *With any of the standard proof systems for first-order logic with equality given in [Troelstra and Schwichtenberg 2000], one can eliminate the propositional connective $\leftrightarrow$ in polynomial time.*

PROOF. By Theorem 2.2, it suffices to show that one can eliminate $\leftrightarrow$ efficiently in the corresponding proof systems with definitions. Use definitions to translate formulae in the language with $\leftrightarrow$ to the language without: translate $\varphi(\vec{w}) \leftrightarrow \psi(\vec{z})$ to $(R_\varphi(\vec{w}) \rightarrow R_\psi(\vec{z})) \wedge (R_\psi(\vec{z}) \rightarrow R_\varphi(\vec{w}))$, where R_φ and R_ψ are defined to be equivalent to the translations of φ and ψ, respectively. It is not hard to justify the translated inferences efficiently. □

3. ELIMINATING SKOLEM FUNCTIONS

The following is the analogue of Definition 2.1 for Skolem functions.

Definition 3.1. Let Γ be a set of first-order sentences in a language L. Say that Γ has an *efficient elimination of Skolem functions* if there is a polynomial-time algorithm that behaves as follows: whenever $f_0(\vec{x}_0), \ldots, f_k(\vec{x}_k)$ are new function symbols of various arities, $\varphi_0(\vec{x}_0, y), \ldots, \varphi_k(\vec{x}_k, y)$ are formulae such that each φ_i is in the language $L \cup \{f_0, \ldots, f_{i-1}\}$, and d is a proof of a formula ψ in L from

$$\Gamma \cup \{\forall \vec{x}_0, y\ (\varphi_0(\vec{x}_0, y) \rightarrow \varphi_0(\vec{x}_0, f_0(\vec{x}_0))), \ldots,$$
$$\forall \vec{x}_k, y\ (\varphi_k(\vec{x}_k, y) \rightarrow \varphi_k(\vec{x}_k, f_k(\vec{x}_k)))\},$$

then on input d the algorithm returns a proof d' of ψ from Γ using only formulae in L.

Once again, the definition implies that $|d'|$ is bounded by a polynomial in $|d|$. Right off the bat, we have the following.

PROPOSITION 3.2. $\{\forall x, y\ (x = y)\}$ *has an efficient elimination of Skolem functions.*

PROOF. Roughly speaking, if c is the only element of the universe, every term can be replaced by c. □

By way of motivation, note that is not hard to show that, say, Zermelo-Fraenkel set theory has an efficient elimination of Skolem functions. Argue as follows. Suppose d is a proof of a formula ψ from the axioms of ZF and some Skolem functions. Let k be a bound on the complexity of the formulae occurring in this proof. In ZF, one can prove that the set of true sentences of complexity at most $k + 1$ is consistent, and hence has a countable model. This countable model has Skolem functions, which can then be used to interpret the proof d.

This example suggests that one way to proceed is to try to determine how little one can get away with in carrying out an internal semantic argument of this kind. The answer turns out to be: very little.

Definition 3.3. Say a set of sentences Γ *codes finite functions (efficiently)* if for each n there are

—a definable element, "$\emptyset_n$";
—a definable relation, "$x_0, \ldots, x_{n-1} \in dom_n(p)$";
—a definable function, "$eval_n(p, x_0, \ldots, x_{n-1})$"; and
—a definable function, "$p \oplus_n (x_0, \ldots, x_{n-1} \mapsto y)$"

such that, for each n, Γ proves

—$\vec{x} \notin dom_n(\emptyset_n)$
—$\vec{w} \in dom_n(p \oplus_n (\vec{x} \mapsto y)) \leftrightarrow (\vec{w} \in dom_n(p) \vee \vec{w} = \vec{x})$
—$eval_n(p \oplus_n (\vec{x} \mapsto y), \vec{x}) = y$
—$\vec{w} \neq \vec{x} \rightarrow eval_n(p \oplus_n (\vec{x} \mapsto y), \vec{w}) = eval_n(p, \vec{w})$,

and such that all the definitions and proofs can be constructed in time polynomial in n.

Of course, the intuition is that elements of the universe are assumed to code finite partial functions p, $\emptyset_n$ is the function that is nowhere defined, $eval_n(p, \vec{x})$ returns the value of p at $\vec{x}$, $p \oplus_n (\vec{x} \mapsto y)$ is the modification of p which maps $\vec{x}$ to y, and so on. One could, more generally, assume that the codes are elements of a definable set; but then nothing is lost by taking the other elements of the universe to code the empty function.

These requirements are not strong ones. For example, any sequential theory of arithmetic (in the terminology of [Hájek and Pudlák 1993; Krajíček 1995; Pudlák 1998]) codes finite functions, since one can take such functions to be sequences of tuples $\langle \vec{x}, y \rangle$. Below I will drop the subscripts n in $\emptyset_n$, dom_n, etc. and I will write $p(\vec{x})$ instead of $eval(p, \vec{x})$. By passing to a definitional extension, we can assume that these are actually given by symbols in the language.

THEOREM 3.4. *Suppose* Γ *codes finite functions. Then* Γ *has an efficient elimination of Skolem functions.*

PROOF. The proof will occupy most of the remainder of this section. By Proposition 3.2 we can assume that there are at least two elements in the universe, and so, by Theorem 2.2, we can use definitions freely. By way of exposition, I will first focus on the case where $k = 0$, i.e. there is only one Skolem function to eliminate. (This part does not require definitions.) Then I will discuss the steps necessary to eliminate multiple, possibly nested instances Skolem functions. (This is the part that requires definitions.)

Suppose we want to eliminate the use of a single Skolem function, with defining axiom $\forall \vec{x}, y\ (\varphi(\vec{x}, y) \rightarrow \varphi(\vec{x}, f(x)))$. Let L_f denote the language $L \cup \{f\}$. I will define a forcing relation in L, for formulae in L_f. I will then show that Γ proves that the Skolem axiom is forced; and that anything in the original language is forced if and only if it is true. Given a proof d of ψ from Γ together with the Skolem axiom, then, Γ proves that ψ is forced, and hence true.

Now for the details. Let the formula $Cond(p)$ in the language L assert that p is a finite approximation to a Skolem function for φ, that is,

$$\forall \vec{x} \in dom(p)\ \forall y\ (\varphi(\vec{x}, y) \rightarrow \varphi(\vec{x}, p(x))).$$

Let t be a term in L_f, and let p be a variable not occurring in t. Inductively we will define a term t^p in the language of L, whose free variables are those of t together with p. Intuitively, t^p is the value of t, when f is interpreted by p. At the same time, we will define a relation "t^p is defined," asserting that the value of t^p makes sense. Let

—$x^p \equiv x$, for each variable x (other than p),
—$(g(t_0, \ldots, t_m))^p \equiv g(t_0^p, \ldots, t_m^p)$, for each function symbol g of L, and
—$(f(t_0, \ldots, t_n))^p \equiv p(t_0^p, \ldots, t_n^p)$.

Define "t^p is defined" inductively as follows:

—"x^p is defined" is always true.
—"$(g(t_0, \ldots, t_m))^p$ is defined," where g is a function symbol of L, is true if and only if $t_0^p, \ldots, t_m^p$ are all defined.
—"$(f(t_0, \ldots, t_n))^p$ is defined" is true if and only if $t_0^p, \ldots, t_n^p$ are all defined and $t_0^p, \ldots, t_n^p \in dom(p)$.

If p and q are conditions, say $p \preceq q$, "p is stronger than or equal to q", if p extends q as a function:

$$\forall \vec{x}\ (\vec{x} \in dom(q) \to \vec{x} \in dom(p) \wedge p(\vec{x}) = q(\vec{x})).$$

Now we can define the relation $p \Vdash \theta$ inductively. We can assume that the language has connectives $\wedge$, $\to$, $\forall$, and $\neg$, with $\exists$ and $\vee$ defined from these in the usual way.

(1) $p \Vdash R(t_0, \ldots, t_m)$ if and only if $\forall q \preceq p\ \exists r \preceq q$ $(t_0^r, \ldots, t_m^r$ are all defined and $R(t_0^r, \ldots, t_m^r))$.
(2) $p \Vdash \theta \wedge \eta$ if and only if $p \Vdash \theta$ and $p \Vdash \eta$.
(3) $p \Vdash \theta \to \eta$ if and only if $\forall q \preceq p\ (q \Vdash \theta \to q \Vdash \eta)$.
(4) $p \Vdash \neg\theta$ if and only if $\forall q \preceq p\ q \nVdash \theta$.
(5) $p \Vdash \forall x\ \theta$ if and only if $\forall x\ p \Vdash \theta$.

The quantifiers involving q and r above are intended to range over conditions, so, for example, $\forall q \preceq p\ \ldots$ abbreviates $\forall q\ (Cond(q) \wedge q \preceq p \to \ldots)$. For each θ, the relation $p \Vdash \theta$ is a formula in the language of L whose free variables are those of θ together with p. Note that the length of $p \Vdash \theta$ can be bounded by a polynomial in $|\theta|$ (as well as in $|\varphi|$, which is being held fixed for the moment).

The phrase "θ is forced" and the notation $\Vdash \theta$ abbreviate $\forall p\ (Cond(p) \to p \Vdash \theta)$. In the lemmata that follow, $p, q, r \ldots$ are assumed to range over conditions. Most of the proofs are routine and standard, modulo the additional notes provided below. It is important to recognize that the all the proofs alluded to in the statement of the lemmata can be constructed in time that is polynomial in the length of the assertion being proved, but having stated this up front, I will not bother to repeat it each time.

LEMMA 3.5 MONOTONICITY. *For each formula θ of L_f, Γ proves*

$$p \Vdash \theta \wedge q \preceq p \to q \Vdash \theta.$$

LEMMA 3.6. *For each formula θ of L_f, Γ proves*

$$p \Vdash \theta \leftrightarrow \forall q \preceq p \; \exists r \preceq q \; r \Vdash \theta.$$

COROLLARY 3.7. *For each formula θ of L_f, Γ proves*

$$\Vdash (\theta \leftrightarrow \neg\neg\theta).$$

LEMMA 3.8. *For any term t of L_f, Γ proves*

$$\forall q \; \exists r \preceq q \; (t^r \textit{ is defined}).$$

PROOF. Use induction on the term t. The only interesting case is where t is of the form $f(s_0, \ldots, s_k)$. By the induction hypothesis, we can find an $r' \preceq q$ such that $s_0^{r'}, \ldots, s_k^{r'}$ are all defined. If $s_0^{r'}, \ldots, s_k^{r'} \in dom(r')$, take $r = r'$. Otherwise, if $\exists y \; \varphi(s_0^{r'}, \ldots, s_k^{r'}, y)$, let $r = r' \oplus (s_0^{r'}, \ldots, s_k^{r'} \mapsto y)$, for any such y; and if $\forall y \; \neg\varphi(s_0^{r'}, \ldots, s_k^{r'}, y)$, let $r = r' \oplus (s_0^{r'}, \ldots, s_k^{r'} \mapsto y)$, for any y at all. □

The next two lemmata are proved by induction on s and θ, respectively.

LEMMA 3.9. *If t and $s(x)$ are any terms of L_f, Γ proves*

$$t^p = z \rightarrow (s(t)^p = s(z)^p)$$

LEMMA 3.10. *If $\theta(x)$ is any formula of L_f and t is any term of L_f then Γ proves*

$$(t^p \textit{ is defined} \wedge t^p = z) \rightarrow (p \Vdash \theta(t) \leftrightarrow p \Vdash \theta(z)).$$

LEMMA 3.11. *For each formula θ of L_f, if θ is provable in classical first-order logic, then Γ proves $\Vdash \theta$.*

PROOF. The proof is for the most part standard and routine, though one has to be a little bit careful with the quantifier axioms and rules since terms might not always be "defined." To show $\forall x \; \theta(x) \rightarrow \theta(t)$ is forced, let us argue in first-order logic from assumptions in Γ. Suppose $p \Vdash \forall x \; \theta(x)$. By Lemma 3.6 it suffices to show $\forall q \preceq p \; \exists r \preceq q \; \theta(t)$. So suppose $q \preceq p$, and by Lemma 3.8 let $r \preceq q$ be such that t^r is defined. Let $z = t^r$. By monotonicity, $r \Vdash \forall x \; \theta(x)$, so $r \Vdash \theta(z)$. By Lemma 3.10, $r \Vdash \theta(t)$. □

A formula in the original language is forced if and only if it is true.

LEMMA 3.12. *For each formula θ of L, Γ proves $(p \Vdash \theta) \leftrightarrow \theta$.*

PROOF. Induction on θ. □

The next lemma is the important one: it asserts that the Skolem axiom is forced.

LEMMA 3.13. *Γ proves $\Vdash \forall \vec{x}, y \; (\varphi(\vec{x}, y) \rightarrow \varphi(\vec{x}, f(\vec{x})))$.*

PROOF. Once again, argue in first-order logic, from Γ. Suppose for some $\vec{x}, y$ we have $p \Vdash \varphi(\vec{x}, y)$. By Lemma 3.12, $\varphi(\vec{x}, y)$. By Lemma 3.6, it suffices to show $\forall q \preceq p \; \exists r \preceq q \; r \Vdash \varphi(\vec{x}, f(\vec{x}))$, so suppose $q \preceq p$. If $\vec{x} \in dom(q)$, the fact that q is a condition guarantees $\varphi(\vec{x}, f(\vec{x}))$, and we can take $r = q$; otherwise, take $r = q \oplus (\vec{x} \mapsto y)$. Either way, as above, we have $r \Vdash \varphi(\vec{x}, f(\vec{x}))$, as required. □

Proof of Theorem 3.4, for a single Skolem function. Suppose there is a proof d of a formula ψ in the language L from finitely many sentences in $\Gamma \cup \{\forall \vec{x}, y\ (\varphi(\vec{x}, y) \to \varphi(\vec{x}, f(x)))\}$. By Lemma 3.11, Γ proves that this implication is forced. By Lemmata 3.12 and 3.13, Γ proves that all the hypotheses are forced, so Γ proves that ψ is forced as well. By Lemma 3.12, Γ proves ψ.

Since each component of the derivation just described can be constructed in time polynomial in $|d|$, so can the entire proof. □

To extend the proof to arbitrary nested definitions of Skolem functions, we need to iterate the forcing definition. A similar iteration was used in [Avigad 1996]; the situation here is easier, since we only have to deal with finite iterations.

Let $d, f_0, \ldots, f_k, \varphi_0, \ldots, \varphi_k$ be as in Definition 3.1. For each $i \leq k$, we will define the notion of an i-condition, an ordering $\preceq_i$ on i-conditions, and a forcing relation $\Vdash_i$ between i-conditions and formulae θ in the language $L \cup \{f_0, \ldots, f_i\}$. An i-condition consists of a sequence $p_0, \ldots, p_i$ of finite functions, with arities corresponding to those of $f_0, \ldots, f_i$. As expected, $p_0, \ldots, p_i \preceq_i q_0, \ldots, q_i$ means that each p_j extends q_j, as above.

The two notions Cond_i and $\Vdash_i$ are defined simultaneously, by recursion on i. $\mathit{Cond}_0(p)$ and $p \Vdash_0 \theta$ are defined as above, in the case where there is only one Skolem function. Assuming Cond_i and $\Vdash_i$ have been defined, the relation $\mathit{Cond}_{i+1}(p_0, \ldots, p_{i+1})$ is defined by

$$\begin{aligned} &\mathit{Cond}_i(p_0, \ldots, p_i) \wedge p_0, \ldots, p_i \Vdash_i \\ &\quad \forall \vec{x}_{i+1}, y\ (\vec{x}_{i+1} \in \mathit{dom}(p_{i+1}) \wedge \varphi(\vec{x}_{i+1}, y) \to \varphi(\vec{x}_{i+1}, p(\vec{x}))). \end{aligned}$$

In the atomic case, assuming that $t_0, \ldots, t_m$ are terms in the language of $L \cup \{f_0, \ldots, f_{i+1}\}$, the relation $p_0, \ldots, p_{i+1} \Vdash_{i+1} A(t_0, \ldots, t_m)$ is defined by

$$\forall \vec{q} \preceq \vec{p}\ \exists \vec{r} \preceq \vec{q}\ (t_0^{\vec{r}}, \ldots, t_m^{\vec{r}} \text{ are defined and } A(t_0^{\vec{r}}, \ldots, t_m^{\vec{r}})).$$

The forcing relation is then extended to arbitrary formulae in the language as above. Notice that the relation $\Vdash_i$ is used in the definition of Cond_{i+1}, which is in turn used to define $\Vdash_{i+1}$. By introducing new relation symbols to represent the definitions of $\mathit{Cond}_0, \ldots, \mathit{Cond}_k$, we can bound the lengths of all the formulae involved by a polynomial.

LEMMA 3.14. *For each $i \leq k$, Lemmata 3.5–3.11 hold for i-conditions, $\preceq_i$, and $\Vdash_i$.*

LEMMA 3.15. *For each $i \leq k$, if θ is in the language $L \cup \{f_0, \ldots, f_i\}$, then Γ proves the following:*

$$p_0, \ldots, p_k \Vdash_k \theta \leftrightarrow p_0, \ldots, p_i \Vdash_i \theta.$$

LEMMA 3.16. *For each $i \leq k$, Γ proves that the ith Skolem axiom is k-forced.*

Once again, the relevant proofs can be constructed in time polynomial in $|d|$. The proof of Theorem 3.4 now follows exactly as in the case of a single Skolem function. □

If a and b are distinct and f is a Skolem function for $(\varphi(\vec{x}) \wedge y = a) \vee (\neg\varphi(\vec{x}) \wedge y =$

b), then $f(\vec{x}) = a$ serves as a definition for $\varphi(\vec{x})$. As a corollary to Theorem 3.4 we have the following:

COROLLARY 3.17. *Suppose* Γ *codes finite functions and proves* $\exists x, y\ (x \neq y)$. *Then one can eliminate arbitrary nested instances of definitions and Skolem functions from proofs in* Γ*, with a polynomial bound on the increase in the lengths of proofs.*

4. QUESTIONS

In standard terminology (e.g. [Krajíček 1995; Pudlák 1998]), Section 2 shows that one can eliminate definitions from proofs in first-order logic in polynomial time if and only if extended Frege systems for propositional logic can be p-simulated by Frege systems. As noted above, whether or not this is the case is still open. Section 2 shows that Theorem 2.2 and Corollary 2.5 hold for first-order logic with equality. What can one say in the absence of equality?

It is also still open as to whether one can efficiently eliminate even a single Skolem function from proofs in pure logic, or from theories that do not code finite functions.

The elimination of definitions in Section 2 used the law of the excluded middle. As a result, it is open as to whether one has an efficient elimination of definitions in intuitionistic first-order logic. (See also [Schwichtenberg 1979] for a discussion of choice functions in the intuitionistic setting.)

ACKNOWLEDGMENTS

I am grateful to Samuel Buss for advice, and to the referees for comments and suggestions.

REFERENCES

AJTAI, M. 1988. The complexity of the pigeonhole principle. In *Proceedings of the IEEE 29th Annual Symposium on Foundations of Computer Science*. 346–355.

AVIGAD, J. 1996. Formalizing forcing arguments in subsystems of second-order arithmetic. *Ann. Pure Appl. Logic 82,* 2, 165–191.

AVIGAD, J. 2000. A realizability interpretation for classical arithmetic. In *Logic Colloquium '98 (Prague)*. Lect. Notes Log., vol. 13. Assoc. Symbol. Logic, Urbana, IL, 57–90.

AVIGAD, J. 2001a. Algebraic proofs of cut elimination. *J. Log. Algebr. Program. 49,* 1-2, 15–30.

AVIGAD, J. 2001b. Eliminating definitions and Skolem functions in first-order logic. In *Proceedings of the 16th annual IEEE symposium on logic in computer science*. 139–146.

AVIGAD, J. 2001c. Weak theories of nonstandard arithmetic and analysis. To appear.

CLOTE, P. AND KRAJÍČEK, J. 1993. *Arithmetic, Proof Theory, and Computational Complexity*. Oxford University Press.

HÁJEK, P. AND PUDLÁK, P. 1993. *Metamathematics of first-order arithmetic*. Springer, Berlin.

HILBERT, D. AND BERNAYS, P. 1939. *Grundlagen der Mathematik*. Vol. II. Springer, Berlin.

KRAJÍČEK, J. 1995. *Bounded arithmetic, propositional logic, and complexity theory*. Encyclopedia of Mathematics and its Applications, vol. 60. Cambridge University Press, Cambridge.

PARIS, J. AND WILKIE, A. 1985. Counting problems in bounded arithmetic. In *Methods in mathematical logic (Caracas, 1983)*. Lecture Notes in Math., vol. 1130. Springer, Berlin, 317–340.

PUDLÁK, P. 1998. The lengths of proofs. In *Handbook of proof theory*. Stud. Logic Found. Math., vol. 137. North-Holland, Amsterdam, 547–637.

SCHWICHTENBERG, H. 1979. Logic and the axiom of choice. In *Logic Colloquium '78*, M. B. et al, Ed. North-Holland, 351–356.

Shoenfield, J. R. 2001. *Mathematical logic*. Association for Symbolic Logic, Urbana, IL. Reprint of the 1973 second printing.

Takeuti, G. 1987. *Proof Theory*, second ed. North-Holland, Amsterdam.

Troelstra, A. S. and Schwichtenberg, H. 2000. *Basic Proof Theory*, second ed. Cambridge University Press, Cambridge.

Received October 2001; revised June 2002; accepted October 2002