A Survey of Some Recent Results on Computational Complexity in Weak Theories of Arithmetic

D.A. Joseph*

P. R. Young**

June 1982

TR 82-501

Department of Computer Science
Cornell University
Ithaca, New York 14853

* Computer Science Department, Cornell University.

** Computer Sciences Department, Purdue University.

This research was supported in part by NSF grant MCS7609232A02, an IBM Graduate Fellowship and an NSF Postdoctoral Fellowship.

A Survey of Some Recent Results

on Computational Complexity

in Weak Theories of Arithmetic

Deborah Joseph

Computer Science Department

Cornell University

Paul Young

Computer Sciences Department

Purdue University

ABSTRACT

In spite of the fact that a great deal of effort has been expended trying to prove lower bounds for algorithms and trying to solve the P = NP question, only limited progress has been made. Although most computer scientists remain convinced that solutions will be found, others (Hartmanis and Hopcroft, Fortune, Leivant and O'Donnell and Phillips) have questioned the adequacy of Peano arithmetic for computer science. This uncertainty has only been increased by the recent work of Paris and Harrington, showing that certain simple, finitistic, combinatorial statements are in fact independent of Peano Arithmetic. In this paper we survey complexity theoretic statements that are known to be independent of arithmetic theories. In addition, we survey recent results analyzing the arithmetic quantifier structure of computational problems.

Keywords: Independence results, NP =?coNP, P =? NP, Peano arithmetic

The report is a revised version of a paper presented at the 10th Symposium on the Mathematical Foundations of Computer Science, Strbske Pleso, Czechoslovakia (Aug. 31 - Sept. 4, 1981).

This work was supported in part by NSF grant MCS-7609232A02, an IBM Graduate Fellowship and an NSF Postdoctoral Fellowship.

1. Introduction

It has been fifty years since Goedel showed that Peano arithmetic is inadequate for a complete analysis and understanding of arithmetic and all its attendant branches of mathematics (including computer science). In the intervening years, mathematicians have come to believe and accept that, in spite of Goedel's work, most arithmetic problems of *real* interest can in fact be handled by Peano arithmetic. Computer scientists, whose mathematical training has largely been from this same school, have thus been convinced that Peano arithmetic provides an adequate formal tool for analysis of their programming languages and problems. However, in spite of the fact that a great deal of effort has been expended trying to prove lower bounds for algorithms and trying to solve the P = NP question, only limited progress has been made. Although most computer scientists remain convinced that solutions will be found, others (Hartmanis and Hopcroft [H&H-76], Fortune, Leivant and O'Donnell [FOR-79], [LEI-81b], [ODO-79], [F,L&O-81], and Phillips [PHI-79]) have questioned the adequacy of Peano arithmetic for computer science. This uncertainty has only been increased by the recent work of Paris and Harrington [PAR-78], [P&H-78], showing that certain simple, finitistic, combinatorial statements are in fact independent of Peano arithmetic. In this paper we survey complexity theoretic statements that are known to be independent of arithmetic theories.

2. Independence Results for Peano Arithmetic

In the early 1900's many mathematicians were interested in obtaining a complete axiom system for all of mathematics. A variety of formal axiom systems were introduced. Despite the fact that Goedel later showed that all of mathematics can not be uniformly axiomatized, many of these formal systems have been very useful for proving theorems in various branches of mathematics. One such formal system is Peano arithmetic. Today it is generally believed that all of the theorems of real interest pertaining to arithmetic and finitary combinatorics can be proved using Peano arithmetic. For this reason most computer scientists believe that Peano arithmetic is an adequate formal system for resolving questions concerning program behavior and complexity[1]. In this paper we will survey results that question whether Peano arithmetic and certain other formal systems that are weaker than Peano arithmetic are adequate for resolving questions of program behavior and complexity. For this reason we will be discussing statments that are *independent* of these formal systems. Recall that a statement is independent of a theory T if it is expressible in the language of T but neither it nor its negation is provable from T. If a statement is independent of a theory T then the theory T is inadequate for resolving its truth or falsity. If natural statements concerning program behavior are independent of Peano arithmetic or various other arithmetic theories, then those theories are inadequate for answering natural questions concerning program behavior.

· Independence results can be proved using two fairly distinct techniques. Historically, independence proofs for arithmetic theories were primarily proof theoretic in nature and involved a detailed arithmetization of the proof theory for the theories. Although many of the more recent results have also been proof theoretic, they tend to rely on results arising from the earlier arithmetizations which established bounds on how rapidly functions could grow and still be provably total. Examples of this sort of approach are the results of Paris and Harrington [P&H-78] showing that an extension of Ramsey's theorem is independent of Peano arithmetic and the results of Fortune, Leivant and O'Donnell [FOR-79], [LEI-81b], [ODO-79], [F,L&O-81], showing that the termination theorem for certain strongly-typed programming languages is independent of Peano arithmetic. A second technique for proving statements independent is model theoretic and involves constructing one model of arithmetic in which the independent statement is true and a second model in which it is false. Considerable work concerning model theoretic techniques for obtaining arithmetic independence results has been done by Kirby and Paris [K&P-76] and is discussed briefly below. Many of the results that we discuss later will be proved using model theoretic techniques.

For background we begin with a brief discussion of some of the more important early

1. The recent results of Paris and Harrington, [P&H-78], [PAR-78], and Fortune, Leivant and O'Donnell, [FOR-79], [LEI-81b], [ODO-79], [F,L&O-81], which we discuss later, cast some doubt on this belief.

independence results for Peano arithmetic.

2.1. Goedel's Theorems

In 1900 Hilbert, [HIL-02], posed the problem of whether a complete axiomatization could be given for mathematics. At that time he and many other mathematicians believed that there should be a sound and complete set of axioms from which all the true statements of mathematics could be derived. In 1931 Goedel showed that this is *not* the case. In his paper [GOD-31], "On formally undecidable propositions of Principia Mathematica and related systems I", Goedel showed that for any theory containing very minimal arithmetic axioms there are sentences that are true but unprovable:

Theorem (Goedel): In every consistent formal system that contains a certain amount of finitary number theory, there exist undecidable propositions [neither they nor their negations are provable]. Moreover, the consistency of any such system cannot be proved within the system.

Although Goedel's theorem dashed all hopes of finding a set of axioms from which all the true theorems of arithmetic could be derived, until very recently no naturally occurring true statements about finitary arithmetic had been found that could not be proven from the Peano axioms, an axiomatic system which is more than adequate for applying Goedel's theorem.

Goedel's proofs of these results were proof theoretic in nature and involved a lengthy arithmetization of the proof theory for arithmetic.

Other independence results that are important for historical reasons are those that follow from the proof of Ryll-Nardzewski, [R-N-52], showing that Peano arithmetic is not finitely axiomatizable. If for no other reason, these results are important because they use purely model theoretic techniques. The proofs construct nonstandard models to show that for any finite class of axioms for Peano arithmetic there are induction axioms that are independent of the axioms.

2.2. Ramsey's Theorem

As mentioned above, the result of Paris and Harrington, [P&H-78], [PAR-78], is startling because it represents perhaps the first example of a naturally occurring, nonmetamathematical, statement about the natural numbers that is true but not provable from Peano arithmetic. The result is that a simple extension of the finite version of Ramsey's theorem, [RAM-29], is independent of Peano arithmetic. The standard finite version of Ramsey's theorem is the following:

Theorem (Ramsey): For all natural numbers e, r, k there is an integer M such that, if all of the e-element subsets of M are partitioned into r disjoint classes then there is a homogeneous subset of M of size at least k. (By homogeneous we mean that all of its e-element subsets were placed in the same r-class.)

The extension to Ramsey's theorem that Paris and Harrington show is independent of Peano arithmetic is obtained by requiring that the homogeneous set be *relatively large,* where relatively large means that the cardinality of the set is greater than its minimal element.

This result has had considerable impact for two reasons: First, it has caused mathematicians *and computer scientists,* whose work is often combinatorial, to question whether other fairly natural combinatorial statements may be independent of Peano arithmetic. Second, Paris' proof, [PAR-78], is essentially model theoretic. Thus the proof has sparked a new interest among mathematicians and computer scientists in the use of nonstandard models. For example, some of the ideas used by Lipton, [LIP-78] arise naturally from the work of Kirby and Paris, [K&P-76], which developed the model theoretic techniques initially used ([PAR-78]) for obtaining Ramsey independence results.

2.3. The Impact of Goedel's Theorem on Computer Science

Independence results in computer science have a history which goes back more than fifteen years (Fischer [FIS-65], Young [YOU-77], Hartmanis [HART-77], Hartmanis and Hopcroft [H&H-76], Gordon [GOR-79], and Phillips [PHI-79]). The notion of a *provably* recursive function was introduced by Kreisel as early as 1952, [KRE-52]. In a somewhat later paper, [KRE-58], Kreisel showed that the recursive functions provably total using Peano arithmetic are exactly the same as those provably total using intuitionistic or Heyting arithmetic [HEY-59]. In [FIS-65] Fischer studied the class of provably recursive functions in detail and compared them with the class of total recursive functions. Somewhat later Young, [YOU-77], studied optimization among provably equivalent programs and showed that questions of speed-up and optimality are dependent, not on the function computed, but instead on the particular program which is chosen to compute the given function.

The first paper that *explicitly* investigated independence results relating to computer science was presented by Hartmanis and Hopcroft, [H&H-76]. They showed that the Goedel incompleteness results for arithmetic could be coded into many problems of interest to computer scientists. Specifically, their paper contains the following results (for F any formal system adequate to prove the Goedel incompleteness results):

Theorem (Hartmanis and Hopcroft): For every F we can effectively construct an i such that ϕ_i is recursive and $P^{\phi_i} = NP^{\phi_i}$ can be neither proved nor disproved in F.

Theorem (Hartmanis and Hopcroft): For every such F, there exists an algorithm (which can be explicitly given) whose running time is n^2, but there is no proof in F that it runs in time $< 2^n$.

Although Hartmanis and Hopcroft's results show that there are statements of apparent interest to computer scientists that are independent of Peano arithmetic, *none* of these early papers relating independence results to computer science produced results that could be described as surprising: All of the results were obtained by straightforward diagonalizations and by coding standard Goedel undecidable sentences into questions of computer science. The resulting programs and problems could hardly be expected to arise in practice or to give insight into real problems. A fact which is nicely illustrated by results of Hajek [HAJ-77], [HAJ-79] and Grant [GRA-80] showing quite clearly that such codings of incompleteness results give little or no insight into the independence of P =? NP from Peano arithmetic.

2.4. The Termination Theorem for Polymorphic Typed Programming Languages

In [ODO-79], O'Donnell presented the first natural example of a statement concerning programming languages that is true but unprovable using Peano arithmetic. This work has subsequently been extended by Fortune [FOR-80], Leivant [LEI-81b], and O'Donnell [F,L&O-80]. They show that for certain strongly-typed programming languages similar to CLU, ALPHARD, MODEL and RUSSELL, the question of program termination for programs which in less sophisticated languages would be trivial straight-line code is independent of second order Peano arithmetic. The results begin by observing the

Termination Theorem: Every program containing only total operations and nonrecursive function definitions and applications is total.

They then go on to prove

Theorem (Fortune, Leivant & O'Donnell): The Termination Theorem for the Polymorphic Typed Programming Language is independent of second order Peano arithmetic.

The implication of this work is twofold: First, if languages become too sophisticated, verification of simple correctness may become literally impossible unless increasingly sophisticated formal systems are available. Second, the theorem reinforces the view expressed earlier by Hartmanis and Hopcroft that Peano arithmetic may be inadequate for solving some of the long-standing open questions of computer science.

The proof of Fortune, Leivant and O'Donnell is proof theoretic and uses the same sort of techniques as those used by Paris and Harrington in [P&H-78]: O'Donnell *et al* show that a universal function for the polymorphic typed programs which contain only total operations and nonrecursive function definitions and applications must grow faster than any function which is provably total in 2nd order Peano arithmetic.

2.5. P =? NP and Models of Complete Arithmetic

Recently Dimitracopoulos and Paris [P&D-80] have used model theoretic techniques to investigating the possible independence of computational statements. They show that the P = NP and NP = coNP questions are related to certain natural problems concerning definability in nonstandard models of complete arithmetic[2]. For example, they show that P = NP if and only if there is a nonstandard model of arithmetic in which elements are "equivalent" whenever they cannot be distinguished by any standard polynomial time program.

3. Complexity Theory and Weak Arithmetic Theories

The independence results surveyed in the preceding section were primarily concerned with Peano arithmetic and its extensions. Computer scientists have recently given considerable attention to the study of formal systems that are weaker than Peano arithmetic. Although it is undoubtedly true that independence results for full Peano arithmetic are of primary importance, weaker fragments of arithmetic seem to be more tractable, and perhaps as a result, there is a rapidly growing body of work relating complexity-theoretic results to such fragments. Most of the systems studied are not rich enough to prove all of the theorems that interest computer scientists. Nevertheless computational complexity theoretic independence results for these systems are of interest for a variety of reasons: (i) for better understanding exactly how powerful proof tools must be in computer science, (ii) for classifying the proof-theoretic complexity of results in computer science, and (iii) as precursors for independence results either for richer axiom systems or for more interesting statements. In addition, for some of these weak theories the independence of specific computational statements is equivalent to their independence from full Peano arithmetic. This is because several of these theories include in their axioms all of the true or provable arithmetic sentences that have very restricted quantifier structure (at most one alternation of quantifiers). Many important open questions of computer science can be expressed by such simple sentences. For example, the formal sentence which asserts that "P = NP" is of the form $(\exists y)(\forall x)S(x,y)$ where S involves only bounded quantifiers. Similarly, the formal sentence that

2. Complete arithmetic is the theory which has as axioms all of the *true* sentences about addition and multiplication on the natural numbers.

asserts that "P $\neq$ NP" is of the form $(\forall x)(\exists y)R(x,y)$ with R involving only bounded quantifiers. Statements asserting the equivalence or inequivalence of NP and coNP also have a fairly simple form. Analyzing the quantifier structure of these computational statements is dependent on obtaining suitable arithmetic characterizations of sets in NP, and considerable work in this area already exists:

3.1. Quantifier Structure of Sets in NP and coNP

Some of the earliest work on arithmetically characterizing sets in the class NP was done by Adleman and Manders [A&M-76]. They showed that:

i) If membership in a set S is expressible by a Diophantine predicate in the form:

$$n \in S \Leftrightarrow (\exists y_1; |y_1| \leq p(|n|)) \cdots (\exists y_k; |y_k| \leq p(|n|))[P_S(n; y_1, \ldots, y_k) = 0]$$

where p and P_S are polynomials, then $S \in$ NP.

ii) If $S \in$ NP, then membership in S can be expressed by the Diophantine predicate:

$$n \in S \Leftrightarrow (\exists y_1; |y_1| \leq 2^{q(|n|)}) \cdots (\exists y_m; |y_m| \leq 2^{q(|n|)})[Q_S(n; y_1, \ldots, y_m) = 0]$$

where q and Q_S are polynomials.

In somewhat later work Stockmeyer [STO-77] and Wrathall [WRA-78] considered the classes of sets definable in the form:

$$n \in S \Leftrightarrow (\{{}^{\forall}_{\exists}\} y_1; |y_1| \leq p_1(|n|)) \cdots (\{{}^{\forall}_{\exists}\} y_k; |y_k| \leq p_k(|n|)) \quad [PL(n, y_1, \ldots, y_k)] \tag{2.1}$$

where the quantifiers alternate and the matrix of the expression, $PL(n, y_1, \ldots y_k)$, is a predicate computable in polynomial time. (Earlier Karp [KAR-72] had observed that the class NP is exactly the class of sets definable in this way using only one existential quantifier.) Although it is not known whether the sets definable by predicates of the form (2.1) form a hierarchy based on numbers of alternating quantifiers, it is known that such a hierarchy has more than one level iff P $\neq$ NP.

Recently, an exact arithmetic characterization of NP has been given by Kent and Hodgson [K&H-80]. They defined a class of predicates that they call the *Exponential Existential Bounded Arithmetical Predicates* and showed that the class NP is exactly the class of sets definable by Exponential Existential Bounded Arithmetical Predicates:

Definition (Kent and Hodgson): A set of natural numbers, S, is said to be definable in *Exponential Existential Bounded Arithmetical* form if and only if,

$$n \in S \Leftrightarrow \Pi\ [(\exists y_1 \leq 2^{p_1(|n|)}) \ldots (\exists y_k \leq 2^{p_k(|n|)}) \\ (\forall z_1 \leq q_1(|n|)) \cdots (\forall z_m \leq q_m(|n|))] \\ [P_S(n, y_1, \ldots, y_k, z_1, \cdots z_m) = 0]$$

where p_i and q_i are polynomials and P_S is a polynomial and the Π in front of the quantifier prefix indicates that an arbitrary permutation of quantifiers is allowed.

A hierarchy of arithmetic predicates called the *Bounded Arithmetic Predicates* that is similar to the *Exponential Existential Bounded Arithmetical Predicates* was introduced by Harrow [HARR-78]:

Definition (Harrow): A set is defined by a *Bounded Arithmetic Predicate* if and only if

$$n \in S \Leftrightarrow (\exists y_1 \leq p_1(n))(\forall y_2 \leq p_2(n)) \ldots (Qy_k \leq p_k(n))(Qy_{k+1} \leq p_{k+1}(n)) \\ [P_S(n, y_1, \ldots, y_{k+1}) = 0]$$

Harrow showed that the *Bounded Arithmetic Predicates* define exactly the same class of sets as the *Rudimentary Predicates* (RUD) studied by Smullyan [SMU-61], by Wrathall [WRA-78], and by others.

Extending these results, Wrathall [WRA-78] and independently, Hicks and Meloud [MEL-79], showed that for a restricted class of the Rudimentary Predicates called the *Positive Rudimentary Predicates* (R^+), $R^+ =$ RUD only if NP = coNP.

The above results show that the sets in R^+, RUD, NP and coNP can all be defined by formulas which have very restricted quantifier structure. Accordingly, one might expect weak theories that restrict the quantifier structure of their axioms to be useful in studying the relationship between these classes. Relationships between such weak theories and the NP =? coNP questions have been investigated by Wilkie and others.

3.2. NP and coNP in Weak Theories of Arithmetic

Wilkie [WIL-81] related the question of whether NP = coNP to independence results for a weak arithmetic theory. He showed that if Matijasevic's theorem, [MAT-70], is provable in $PA^- + I\Sigma_0$ (Peano arithmetic with induction restricted to bounded formulas), then NP = coNP. The proof uses Adleman and Manders' result showing that the set of equations of the form

$$a\cdot x^2 + b\cdot y = c, \quad a,b,c > 0,$$

solvable on the natural numbers is NP-complete. Manders [MAN-81] has observed that the result can be strengthened to the system $PA^- + I\Sigma^*$ where $I\Sigma^*$ is the induction schema for formulae with bounded quantification of the form

$$(\forall x<2^{|y|^n}); \quad (\exists x<2^{|y|^n}), \text{ for } n \in N.$$

A more model theoretic approach to the NP = coNP problem was taken by DeMillo and Lipton [D&L-79]. They defined a weak theory of arithmetic which they called the Theory of Polynomial Time (PT) and showed that PT is rich enough to verify that NP = coNP iff P = NP:

Definition (DeMillo and Lipton): The language of PT includes symbols for all of the functions and predicates which are computable in polynomial time. The axioms of PT are all true sentences of the form $(\exists x)(\forall y)A(x,y)$, where A is quantifier free.

Definition (DeMillo and Lipton): A theory T is said to *verify* that NP = coNP if for each $S \in$ NP, $T \vdash S \in$ coNP.

Theorem (DeMillo and Lipton): PT can *verify* NP = coNP if and only if P = NP.

On close investigation one discovers that this result rests primarily on the fact that the only functions a theory such as PT can prove total are the functions defined by the terms or those defined by cases from the terms. For PT these functions are essentially the standard polynomial time computable functions since specific symbols for each of these functions were explicitly added to the language of PT.

A similar result relates the theory PT to the question of whether or not sets in NP $\cap$ coNP are decidable in polynomial time:

Definition (DeMillo and Lipton): Let S be a fixed recursive set and let $A(x,y)$ and $B(x,y)$ be defined as follows:

$$(\exists y)A(x,y) \Leftrightarrow x \in S \quad \& \quad (\exists y)B(x,y) \Leftrightarrow x \; nomem \; S.$$

Now let

$$\Delta_S(A,B) = (\forall x)[(\exists y)A(x,y) \vee (\exists z)B(x,z)].$$

Theorem (DeMillo and Lipton): Let $S \in NP \cap coNP$. Then the following are equivalent:

$S \in P$

$PT \vdash \Delta_S(A,B)$ *for some A, B in the language of PT.*

Again, the proof of this theorem has little to do with NP or coNP, but rather depends on the fact that essentially the only sets provably decidable using PT are the standard polynomially decidable sets.

The results discussed in this section show that questions of provability in weak theories of arithmetic are related to computational questions such as whether P = NP and whether NP = coNP. However questions of which computational problems are resolvable in weak theories of arithmetic and which complexity theoretic statements are independent of such theories, are of interest in their own right. Explicit independence results and examples of complexity theoretic statements which *are* unprovable have been obtained for some weak theories, and we survey some of these results in the remaining two sections of this paper.

3.3. The Theory ET

One of the weakest fragments of arithmetic that has been used in studies of computational complexity is the theory ET introduced by DeMillo and Lipton. The theory ET has as axioms all of the true universal sentences from the first order language,

$$L(ET) = \{0,1,+,*,-, \max(x,y), \min(x,y), 2^x, 3^x,..., P_0(x), P_1(x), P_2(x),...\}$$

where the P_i are new predicate symbols corresponding to each of the standard polynomially time computable relations. (By a universal sentence, we mean a sentence of the form $\forall x_1 \forall x_2 \cdots \forall x_n S(x_1,...,x_n)$ where S contains *no* quantifiers, either bounded or unbounded.)

DeMillo and Lipton add a new constant symbol *a* to the language of ET and define a new class of predicates, which we call $P_{D\&L}$, as follows:

(i) $P_i \in P_{D\&L}$ for all i,

(ii) $P_{D\&L}$ is closed under Boolean operations, and

(iii). If $A(x) \in P_{D\&L}$, then so does $(\exists x < a)A(x)$.

Obviously, in the *standard* model, N (the standard natural numbers), the constant symbol *a* can only be interpreted as a *standard* natural number, and of course each P_i can be correctly interpreted as the polynomially time testable relation which it is supposed to represent. Thus, *in the standard model,* for every interpretation of the constant symbol *a,* all of the predicates in $P_{D\&L}$ are polynomially time testable. Furthermore, if we take any NP-complete set, for example SAT (the set of satisfiable Boolean formulas in conjunctive normal form) then it can be expressed as

$$x \in SAT \quad \text{iff } (\exists y)S(x,y)$$

where S is a polynomially time testable predicate. Therefore in L(ET) there is an explicitly introduced predicate symbol P_S that represents S. With this as motivation, we can now state the primary result of DeMillo and Lipton:

Theorem (DeMillo and Lipton [D&L-80]): Let A be a quantifier free predicate in the language L(ET) such that $\{ x \mid (\exists y)A(x,y)\}$ is NP-complete. Then for some predicate $\phi(x) \in P_{D\&L}$,

$$ET + (\forall x)[(\exists y)A(x,y) \Leftrightarrow \phi(x)]$$

is consistent.

Before applying this theorem to NP-complete sets, one should bear in mind the fact, mentioned earlier, that the sets in NP can be characterized as those sets definable by using Kent-Hodgson Exponential Existential Bounded Arithmetical Predicates. This characterization, along with one's general intuitions, makes it seem unlikely that many (or even any) of the NP-complete sets can be defined in any *natural*[3] way by arithmetical predicates that do not have bounded-quantifiers. On the other hand, in the extended language L(ET) there is an explicitly introduced predicate symbol P_R for *every* polynomially time testable relation R. Therefore, for every NP-complete set such as SAT, one can substitute the associated predicate symbol P_S which is in L(ET) in order to get the desired quantifier free predicates, A, used in the above theorem. Since, in the standard model, all of the predicates, ϕ, in $P_{D\&L}$ are polynomially time testable, this theorem can thus be interpreted as saying that it is consistent with the axioms of ET to believe that P = NP.

Although such an interpretation may be tempting, it is not entirely reasonable. Even in the language of Peano Arithmetic there are obviously many different formal sentences which assert that P = NP. Even for different *natural* sentences which assert that P = NP, it is not clear that in theories as weak as ET that one can prove their equivalence. Additional difficulties arise with the predicate symbols P_i. One such difficulty is the following: Although the axioms of ET are strong enough to guarantee that polynomial time testable predicates like S and their corresponding predicate symbols P_S agree in the standard model N, (since each true instance, e.g., $P_S(3)$ or $\neg P_S(3)$ is an *axiom* of ET), arithmetical predicates like S may involve so much bounded quantification that the axioms of ET cannot guarantee that S and P_S agree in all models of ET. In particular, it is not clear that there is any model of ET in which

$$(\forall x)[(\exists y)P_S(x,y) \Leftrightarrow \phi(x)]$$

holds and at the same time P_S agrees with S, which is the predicate it is supposed to represent; i.e., it is not clear that there are any models of ET in which the preceding formula holds and { x |

3. Matijesevic's Theorem tells us that there is a characterization that does not not involve bounded quantifiers but not necessarily a natural one.

$(\exists y)P_S(x,y)$ } "correctly" represents the set of satisfiable boolean expressions in conjunctive normal form.

Even setting this aside, there are additional difficulties in interpreting what P = NP *means* in models of theories as weak as ET:

First, we observe that the proof of DeMillo and Lipton's Theorem does not use the fact that the set $\{ x \mid (\exists y)A(x,y)\}$ is NP-complete. It only uses the fact that the relation A is expressed by a quantifier-free formula in the language L(ET). Since every recursively enumerable set is characterized by *some* existential predicate (again, this is Matijesevic's Theorem), it is just as reasonable to assert that it is consistent with ET to believe that *every recursively enumerable set* is testable in polynomial time as it is to assert this merely of every NP-complete set.

Second, it is not clear that in the models of ET in which

$$(\forall x)[(\exists y)A(x,y) \Leftrightarrow \phi(x)]$$

holds for $\phi \in P_{D\&L}$ that all of the members of $P_{D\&L}$ are in fact testable in polynomial time. In fact, it is not even clear in models of such weak theories what it *should* mean to say that a set is testable in polynomial time. Here again, there are several difficulties. It seems that in theories like ET, one may not even be able to prove that different notions of computation, for example Turing machines, Markov algorithms, or Loop programs are equivalent, or if they are equivalent that they are polynomially time related. Thus, it seems that one must simply pick some notion of computation and analyze that notion.

Even when one has chosen such a notion, for example, the Loop programs of [B&L-74], once one picks a predicate T which encodes sequences of computations, the T-predicate involves so much bounded quantification that it is difficult to know how it behaves in models of theories like ET.[4] However as we observed above, there is a reasonable T-predicate that is testable in polynomial time, so it has a corresponding predicate *symbol,* P_T, explicitly introduced in the language L(ET). Since P_T and T must agree on all of the elements of N, one *might* use P_T as one's

4. As pointed out by Csirmaz ([CSI-80]), similar problems arise for theories lacking multiplication.

definition of computation in models of ET. Although T and P_T will agree on N, T and P_T suffer the same difficulties as S and P_S discussed above. In particular, we cannot guarantee that they agree in nonstandard models of theories as weak as ET. Nevertheless, since the *symbol* P_T obviously involves no bounded quantification, it is tempting to use it for analyzing computations in models of ET. Doing so has shown that with *this* notion of computation:

Theorem (Joseph and Young [J&Y-81]): For every recursively enumerable set $\{ x \mid (\exists y)A(x,y)\}$ there is a formula $\phi(x) \in P_{D\&L}$ for which it is consistent with ET to believe

$$(\forall x)[(\exists y)A(x,y) \Leftrightarrow \phi(x)]$$

and that all formulas ϕ of $P_{D\&L}$ have Loop programs which:

(a) correctly compute ϕ when they halt, and

(b) have a polynomial which bounds the number of steps of each computation.

Unfortunately, even if one accepts P_T as expressing a meaningful notion of computation, the preceding theorem does not guarantee polynomial time *decidability*. For example it is known that in models of weak fragments of arithmetic there can exist programs which run for only a bounded number of steps but still do not halt (Joseph and Young [J&Y-80]). Whether this or some other malady plaques the computations described in the preceding theorem, we are not sure, but nevertheless we can prove:

Theorem (Joseph and Young [J&Y-81]): Under the same conditions as the preceding theorem, in the same model of ET as constructed there; there are members, ϕ, of $P_{D\&L}$ which are *not* decided by any loop program in the model.

Exactly *which* predicates of $P_{D\&L}$ can be tested in polynomial time in interesting nonstandard models of theories like ET is an open question. In spite of this and our uncertainties about computations in models of theories like ET, we do know that the study of fast algorithms in such models is closely tied to how computations behave in the real world. There are theories very similar to ET for which, if there exist models in which SAT *is* decidable in polynomial time, then *in the standard model* of computation SAT has arbitrarily long initial segments which can be decided by short, fast programs. If such short, fast programs for SAT do exist, then from a practical standpoint one might believe that SAT is "almost" polynomially decidable. The following results have been obtained:

Theorem (Joseph and Young [J&Y-81]): It is consistent with ET(Elem), (a theory which is quite similar to ET), to believe that NP-complete sets are polynomially decidable iff in the standard model of computation they have long initial segments that can be decided by short, fast programs.

We certainly believe that NP-complete sets do not have long initial segments which are easily decidable by short programs, but the proof of this seems quite difficult and appears to be closely related to the question of whether or not they have small circuit complexity. We do know however that P-complete sets (with respect to log space reducibility) and EXP-time-complete sets (with respect to polynomial time reducibility) do *not* have long initial segments that are easily decidable by small programs and hence:

Theorem (Joseph and Young [J&Y-81]): It is *not* consistent with ET(Elem) to believe that polynomially complete sets are linearly decidable or that EXP-time-complete sets are polynomially decidable.

3.4. Basic Number Theory

One theory which is strong enough to support unambiguous notions of computation is Basic Number Theory (B). This theory is obtained by taking as axioms all of the Π_2 sentences (sentences of the form $(\forall x)(\exists y)A(x,y)$ where the matrix A is permitted *bounded* quantifiers only) which are provable in Peano Arithmetic. This theory has been extensively studied by logicians (Goldrei, Macintyre, and Simmons [G,M&S-73], Hirschfeld and Wheeler [H&W-75], Macintyre and Simmons [M&S-75], and Simmons [SIM-76]), and a great deal is known about its models. Since Lipton [LIP-78] first suggested its usefulness in studies of computational complexity, its *adequacy* as a basis for theoretical computer science has been a matter of some debate. This issue has been thoroughly discussed elsewhere, and we shall forego a thorough discussion of the adequacy of Basic Number Theory, instead referring the interested reader to such discussion already in the literature (Lipton [LIP-78], Joseph and Young [J&Y-80], and Leviant [LEI-79]). These authors do point out that Basic Number Theory and closely related theories are adequate for proving many standard theorems of mathematics and computer science, a point which has been made much earlier, for example by Rogers [ROG-67], Chapter 14 (and in particular page 322).

For our current purposes, it is enough to know that for models of B notions of computation are invariant: Within B one can prove not only the equivalence of all of the standard models of computation, but also that they are polynomially related. That is, one can prove that the class of functions computable by Loop programs, Turing machines, RAMs or Markov algorithms is exactly the class of partial recursive functions and that the computation times are polynomially invariant. Therefore within models of B a function is easily computable if and only if there is a Loop program which easily computes it. However, as we shall see, in spite of the power of Basic Number Theory, it lacks sufficient inductive power to prove a number of standard results of theoretical and practical computer science.

One of the first computational independence results for weak theories of arithmetic was proved by Lipton [LIP-78] who showed that Rabin's Theorem is independent of Basic Number Theory. It is rather easy to prove that there are recursive sets which have no easy (e.g., linear or polynomial) decision procedures. In fact such a proof can be carried out within Basic Number Theory. Rabin [RAB-60] proved a somewhat stronger theorem, namely that there are recursive sets, S, for which every decision procedure takes a long time (e.g., exponential or more) on *all* but finitely many arguments. It is easy to see that for any such set S, neither S nor $\bar{S}$ can have an easy to decide subset. It is also easy to see that for any set, S, whose characteristic function can be sped-up almost everywhere in the sense of Blum [BLU-67], neither S nor $\bar{S}$ can have an easy to decide subset. In 1978, Lipton showed that these results are independent of Basic Number Theory:

Theorem. (Lipton [LIP-78]): For every standard (provably) recursive set S, there is a model of B in which:

(i) Either S or $\bar{S}$ has a infinite easily decidable subset.

Therefore

(ii) Rabin's theorem on the existence of almost everywhere difficult sets cannot be proven in B,

and

(iii) Blum's theorem on the existence of functions which can be "sped-up" almost everywhere cannot be proven in B.

These results have been strengthened by Leviant:

Theorem (Leviant [LEI-79]): The proceeding results (i)-(iii) apply not just to provably recursive sets S and the theory B, but also to all recursive sets S and to the stronger theory T_{Π_2} which has as axioms all *true* Π_2 sentences.

The two preceding theorems show that some esoteric(?) results of computational complexity cannot be proved within Basic Number Theory and the even stronger theory T_{Π_2}. Some of our own work shows that, despite the fact that Basic Number Theory and its extension T_{Π_2} are fairly powerful theories, these theories possess even more fundamental limitations as formal theories for analyzing program behavior:

Theorem (Joseph and Young [J&Y-80]): There is a fairly large subclass, *S,* of the hypersimple sets (all of which are recursively enumberable but in fact undecidable) and there are nonstandard models of T_{Π_2} in which:

(i) All members of *S* are decidable in linear time.

(ii) Bounded sets are not all decidable.

And

(iii) There are Single Loop programs which contain only subroutine calls and assignment statements, which do not terminate despite the fact that each assignment statement and subroutine call correctly terminates. (Recall that Single Loop programs contain no nested loops and execute each loop only a fixed (constant) number of times.)

These results show that Basic Number Theory and the theory T_{Π_2} are inadequate for distinguishing between undecidability and linear time decidability even for a fairly natural subclass of standard undecidable sets, and that these theories are also inadequate for proving fairly routine statements about program termination for very simple classes of programs.

References

1. Adleman, L. and Manders, K., "Computational complexity of decision procedures for polynomials," Proceedings of the 17th Annual IEEE Symp on Switching and Automata Theory, pp. 169-177 (1976).

2. Baker, T. P., "On 'provable' analogs of P and NP," Math Systems Theory Vol. 12 pp. 213-218 (1979).

3. Berman, L. and Hartmanis, J., "On the isomorphism and density of NP and other complete sets," SIAM J Comput Vol. 6 pp. 305-322 (Dec 1977).

4. Blum, M., "A machine independent theory of the complexity of the recursive functions," J Association Comput Mach Vol. 14(2) pp. 322-336 (1967).

5. Brainerd, W. and Landweber, L., Theory of Computation, John Wiley & Sons, New York (1974).

6. Csirmaz, L., "Structure of program runs of non-standard time," Acta Cybernetica Vol. 4(4) pp. 325-331 (1980).

7. Csirmaz, L., "Programs and program verification in a general setting," Theor Comput Sci, (to appear).

8. DeMillo, R. A. and Lipton, R. J. , "Some connections between mathematical logic and complexity theory," Proceedings of the 11th Symposium on the Theory of Computation, pp. 153-159 (1979).

9. DeMillo, R. A. and Lipton, R. J. , "The consistency of 'P = NP' and related problems with fragments of number theory," Proceedings of the 12th Symposium on the Theory of Computing, pp. 45-57 (May 1980).

10. Dimitracopolous, D. and Paris, J., "Truth definitions for delta-zero formulas," Preprint, (1980).

11. Fischer, P. C., "Theory of provably recursive functions," Trans Amer Math Soc Vol. 117(5) pp. 494-520 (May 1965).

12. Fortune, S., Topics in Computational Complexity, Cornell University (1979). (PhD Dissertation)

13. Fortune, S., Leivant, D., and O'Donnell, M., "The expressiveness of simple and second order type structures," IBM Research Report #RC 8542, (1980).

14. Goldrei, D. C., Macintyre, A., and Simmons, H., "The forcing companions of number theories," Israel J Math Vol. 14 pp. 317-337 (1973).

15. Gordon, D., "Complexity classes of provably recursive functions," J Comput and System Sci Vol. 18(3) pp. 294-303 (June 1979).

16. Grant, P. W., "Some more independence results in complexity theory," Theor Comput Sci Vol. 12 pp. 119-126 (1980).

17. Hajek, P., "Arithmetical complexity of some problems in computer science," pp. 282-287 in Mathematical Foundations of Computer Science, ed. J. Gruska, Springer-Verlag (1977).

18. Hajek, P., "Arithmetical hierarchy and complexity of computation," Theor Comput Sci Vol. 8 pp. 227-237 (1979).

19. Harrow, K., "The bounded arithmetic hierarchy," Inf and Control Vol. 36 pp. 102-117 (1978).

20. Hartmanis, J. , "Relation between diagonalization, proof systems and complexity gaps," Proceedings of the 9th Symposium on the Theory of Computing, pp. 223-227 (1977).

21. Hartmanis, J. and Hopcroft, J. E., "Independence results in computer science," SIGACT News Vol. 8(4) pp. 13-24 (1976).

22. Hirschfeld, J. and Wheeler, W. H., Forcing, arithmetic, and division rings, Springer-Verlag Lecture Notes in Mathematics #454, Berlin & New York (1975).

23. Homer, S. and Reif, J. H., "The complexity of provable properties of first order theories," Preprint, (May 1980).

24. Joseph, D., On the power of formal systems for analyzing linear and polynomial time program behavior, Purdue University (1981). (Ph.D. Dissertation)

25. Joseph, D. and Young, P., "Fast programs for initial segments and polynomial time computation in weak models of arithmetic (preliminary abstract)," Proceedings of the 13th Symposium on the Theory of Computing, pp. 55-61 (May 1981).

26. Joseph, D. and Young, P., "Independence results in computer science?," J Comput Systems Sci, (to appear). (preliminary version in Proc. 12th SIGACT Symp Theor Comput, 1980)

27. Karp, R., "Reducibility among combinatorial problems," in Complexity of Computer Computations, ed. Miller and Thatcher,Plenum Press (1972).

28. Kent, C. and Hodgson, B., "An arithmetical characterization of NP," Lakehead Univ Technical Report, #6-80, (1980). (Dept of Mathematics, Thunder Bay, Ontario, Can.)

29. Kirby, L., Private communication

30. Kirby, L. and Paris, J. , "Initial segments of models of Peano's axioms," in Proceedings Conference Set Theory & Hierarchy Theory V; Bieutowice, Poland, Springer-Verlag Lecture Notes in Mathematics #619, Berlin & New York (1976).

31. Leivant, D., "The complexity of parameter passing in polymorphic procedures," Proceedings of the 13th Symposium on the Theory of Computing, pp. 38-45 (May 1981).

32. Leivant, D., "Unprovability of theorems of complexity theory in weak number theories," Theor Comput Sci, (to appear).

33. Lipton, R. J., "Model theoretic aspects of complexity theory," Proceedings of the 19th Symposium of the Foundations of Computer Science, pp. 193-200 (1978).

34. Macintyre, A. and Simmons, H., "Algebraic properties of number theories," Israel J Math Vol. 22 pp. 7-27 (1975).

35. Mahaney, S., "Sparse complete sets for NP: Solution of a conjecture of Berman and Hartmanis," Proc. of the 21st Symposium on the Foundations of Computer Science, pp. 54-60 (1980).

36. Manders, K., "Computational complexity of decision problems in elementary number theory," in Proceedings of the Set Theory and Hierarchy Theory Conference, Springer-Verlag, Karpacz (1979). (Lecture Notes in Mathematics Series.)

37. Matijesevic, Y., "Enumerable sets are diophantine," Dokl Akad Nauk SSSR Vol. 191 pp. 279-282 (1970).

38. McAloon, K.(ed.), Modeles de l'Arithmetique, Societe Mathematique de France, Paris (1980).

39. O'Donnell, M., "A programming language theorem which is independent of Peano Arithmetic," Proceedings of the 11th Symposium on the Theory of Computation, pp. 176-188 (1979).

40. Paris, J. and Wilkie, A., "Models of arithmetic and rudimentary sets," Preprint, (1980).

41. Paris, J. B., "Some independence results for Peano Arithmetic," J Symbolic Logic Vol. 43 pp. 725-730 (Dec. 1978).

42. Paris, J. B. and Harrington, L., "A mathematical incompleteness in Peano arithmetic," in Handbook of Mathematical Logic, ed. J. Barwise, North-Holland, New York (1978).

43. Phillips, N. C. K., "The P = NP question and recent independence results," Quaestiones Informaticae Vol. 1 pp. 26-29 (June 1979).

44. Rabin, M., "Degree of difficulty of computing a function and a partial ordering of recursive sets," Technical Report Hebrew Univ, (1960).

45. Rogers, H., The Theory of Recursive Functions and Effective Computability, McGraw-Hill, New York (1967).

46. Simmons, H., "Existentially closed models of basic number theory," pp. 325-369 in Logic Colloquium 76, ed. R. Gandy and J. M. R. Hyland, North-Holland (1977).

47. Smullyan, R., "Theory of formal systems," in Annals of Mathematics no. 47, Princeton Univ Press (1961).

48. Stockmeyer, L., "The polynomial-time hierarchy," Theo Comput Sci Vol. 3 pp. 1-22 (1977).

49. Wilkie, A., to appear in Proceedings of the Set Theory and Hierarchy Theory Conference, Springer-Verlag, Karpacz (1979). (Lecture Notes in Mathematics Series.)

50. Wilkie, A., "Applications of complexity theory to sigma-zero definability problems in arithmetic," Preprint, (1980).

51. Wrathall, C., "Rudimentary predicates and relative computability," Siam J Comput Vol. 7 pp. 194-209 (1978).

52. Young, P., "Optimization among provably equivalent programs," J Assoc Comput Mach Vol. 24(4) pp. 693-700 (1977). (Prelim abstract in IEEE SWAT Proc 1973).