JOURNAL OF COMPUTER AND SYSTEM SCIENCES 16, 168–184 (1978)

NP-Complete Decision Problems for Binary Quadratics*

KENNETH L. MANDERS

Group in Logic and Methodology of Science, University of California, Berkeley, California 94720

AND

LEONARD ADLEMAN†

Department of Applied Mathematics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

Received November 24, 1976; revised November 1, 1977

The computational complexity of deciding whether a polynomial with integer coefficients has natural-number zeros ranges from deterministic polynomial time feasibility (for polynomials in one variable or of degree one) to undecidability (presently known to hold for polynomials in nine or more variables). We show that for the two-variable quadratics of the form $\alpha x^2 + \beta y - \gamma = 0$; $\alpha, \beta, \gamma \in \omega$, the problem is *NP*-complete. This implies *NP*-completeness of certain questions about the solutions x of $x^2 \equiv \alpha$ modulo β; $\alpha, \beta, x \in \omega$. It also shows that a nondeterministic Turing machine restricted to evaluating deterministically polynomials of a given form at nondeterministically constructed argument values (called a nondeterministic Diophantine machine below) can solve an *NP*-complete problem in polynomial time.

1. INTRODUCTION

Many number-theoretical questions are formulated or can be formulated as questions about the solvability of Diophantine equations (i.e., $p(x_1 \cdots x_n) = 0$ for a multivariable polynomial $p(x_1 \cdots x_n)$ with integer coefficients) in natural numbers or integers. This becomes clear upon examination of the section on Diophantine problems in any standard number theory text; it is in fact made mathematically precise by Matijasevic's theorem [11] that all recursively enumerable sets are Diophantine (i.e., the elements of the set S correspond to the parameter values a for which an appropriate Diophantine equation $p_S(a, x_1 \cdots x_n) = 0$ has a solution). It is therefore of fundamental importance to consider algorithms for deciding the solvability of Diophantine equations, as Hilbert stressed in

* Research supported by National Science Foundation Grant DCR72-03725-A02.

† While participating in this research, the second author was affiliated with: Computer Science Division, Department of Electrical Engineering and Computer Sciences and the Electronics Research Laboratory, University of California, Berkeley, Calif. 94720.

0022-0000/78/0162–0168$02.00/0

asking for such algorithms in the 10th of his famous set of mathematical problems in 1900 [8]. One wishes to know (a) the complexity of deciding solvability for various classes of Diophantine equations, and especially (b) for which subclasses of Diophantine equations a feasible (i.e., deterministic polynomial time) algorithmic procedure to decide solvability exists.

It follows from Matijasevic's theorem that for some fixed n, the set of solvable n-variable Diophantine equations is nonrecursive. Much effort has been devoted to determining the minimum n for which this is true. The best published result is $n \leqslant 13$ [12]; Matijasevic has improved this to $n \leqslant 9$ [13]. In [12], it is conjectured that $n = 3$ may be possible, though this cannot be shown by present methods.

Even for equations with two unknowns, the decision problem is tantalizingly difficult. For example, the first major positive contribution after Hilbert's address [8] was Thue's theorem [18] (1909) that for any polynomial $f(x, y)$ irreducible over the rationals and homogeneous of degree $\geqslant 3$,

$$f(x, y) - m = 0$$

has at most finitely many integer solutions. Regrettably, Thue's argument provided no algorithm for deciding the solvability of equations of this form, and no such algorithm was found until Alan Baker's fundamental contribution [4] (1967) which yields a nondeterministic exponential time algorithm. Baker's methods extend to various equations in two unknowns [5].

On the other hand, the question of solvability (in integers or natural numbers) of linear equations in two unknowns can be answered in deterministic polynomial time, using the Euclidean algorithm. Thus only the question for binary quadratics remains in the gap between tractable (binary linear) and apparently intractable (binary of degree 3) solvability questions. We show below that for binary quadratics of the form

$$\alpha x_1^2 + \beta x_2 - \gamma = 0, \qquad \alpha, \beta, \gamma \in \omega,$$

the problem of deciding whether there are natural-number solutions is *NP*-complete. The reduction also shows that the problem of deciding whether, for $\alpha, \beta, \gamma \in \omega$,

$$x^2 \equiv \alpha \text{ modulo } \beta, \qquad 0 \leqslant x \leqslant \gamma$$

has a solution in natural numbers in *NP*-complete, even if the prime factorization of the modulus β is given. If the restriction on the size of x is omitted, this problem can be solved in deterministic polynomial time using the factorization of β. (See [7, Sect. 8.4].)

The problems which we show *NP*-complete are distinctly number-theoretic in character; they also differ from the classically *NP*-complete problems (such as propositional satisfiability) with respect to the number of variables involved: Most known *NP*-complete problems involve an unbounded number of variables, whereas our problems involve only two variables. Because of these properties, we hope that the *NP*-completeness of these problems will play a role in showing the *NP*-completeness of further problems of a numerical nature, much as the propositional satisfiability problem has played in showing the *NP*-completeness of combinatorial problems [6, 9].

Our results also give information about a different type of question: What is lost in computational ability and efficiency of a nondeterministic Turing machine if we restrict it to deterministically evaluating polynomials of a fixed form at a nondeterministically constructed argument value, and accepting if and only if the polynomial evaluates to 0? This question is of interest because this computational model is number-theoretically convenient; the sets accepted by such machines are exactly the Diophantine sets. In Section 3 below we consider this model of computation, called a nondeterministic Diophantine machine (NDDM). We give an overview of what is known about the computational ability of NDDM's and relate the results of the present paper to this question. All known results support the conjecture that nondeterministic computation can be studied without loss of generality on NDDM's; this possibility suggests many research questions.

The *NP*-completeness results are formulated and proved in Section 2; Section 3 is devoted to the discussion of NDDM's. Research problems are formulated at the ends of both sections.

2. Main Results

We recall the following (well-known) definitions: A relation R on the natural numbers is *accepted in polynomial time* by a (deterministic or nondeterministic) Turing machine M if and only if there is a polynomial $q(\cdot)$ such that for any $x_1, \ldots, x_n \in \omega$

$$\langle x_1 \cdots x_n \rangle \in R \Leftrightarrow \text{there is a computation of } M \text{ on input } \langle x_1 \cdots x_n \rangle \text{ which halts in an accepting state within } q(|\,\mathbf{x}\,|) \text{ steps, where } |\,x\,| \text{ is the length of } \mathbf{x} \text{ in binary, } \mathbf{x} = \langle x_1 \cdots x_n \rangle.$$

P is the collection of relations on the natural numbers accepted by some deterministic Turing machine (DTM) in polynomial time; NP is the collection of relations on the natural numbers accepted by some nondeterministic Turing machine (NDTM) in polynomial time; a set S in NP is *NP-complete* if for any relation $R \in NP$ there is a deterministic polynomial time computable function $f(\cdot)$ such that

$$(\forall \mathbf{x})[\mathbf{x} \in R \Leftrightarrow f(\mathbf{x}) \in S].$$

Theorem 1. *The (problem of accepting the) set of Diophantine equations (in a standard binary encoding) of the form*

$$\alpha x_1^2 + \beta x_2 - \gamma = 0; \qquad \alpha, \beta, \gamma \in \omega,$$

which have natural-number solutions x_1, x_2 is NP-complete.

Theorem 2. *The (problem of accepting the) set of quadratic congruences (in a standard encoding)*

$$x^2 \equiv \alpha \text{ modulo } \beta$$

with solutions $x \in \omega$ *satisfying*

$$0 \leqslant x \leqslant \gamma; \qquad \alpha, \beta, \gamma \in \omega,$$

is NP-complete.

Remark. As the natural strategy in trying to solve either problem is to factor β and look at congruences modulo prime power factors of β, one might surmise that the difficulty of the problem is in part that of factoring β. This is not so: the proof of Theorems 1 and 2 will show that the problems are still *NP*-complete when β is given in fully factored form.

Theorems 1 and 2 are obtained by a common argument. Let S be the set of satisfiable propositional formulas in conjunctive normal form with at most 3 literals per clause. By Cook [6] it suffices to show that there is a deterministic polynomial-time algorithm which reduces the problem of membership in S to a problem of the form(s) mentioned in the theorem, and that the problems themselves are in *NP*. Both problems considered are solvable by a nondeterministic "guess a solution and check whether it is correct" algorithm in polynomial time, and hence in *NP*. This is because, as is easily verified, there is a bound on the size of possible solutions to either problem given by a polynomial in the coefficients α, β, γ (except in the trivial case when α or β is 0).

We now give the reduction algorithm, followed by proof of correctness and analysis of computation time. The reader may wish to merely skim the algorithm initially, referring back to it while reading the proof and analysis.

The reduction algorithms for Theorems 1 and 2 are identical except for the final step, which will be given separately. The initial steps of the algorithm in fact give a reduction of 3-satisfiability to a convenient special case of knapsack (see [9]). The basic idea behind the algorithm is to set up means of going back and forth between the representation of a sequence as the digits of a number in some base, and as the residues of a number with respect to a system of moduli. (See comment (1) following the proof.)

2.1. *The Algorithm*

"On input ϕ, read ϕ and eliminate all duplicate conjuncts and those in which, for some variable x_i, both x_i and $\bar{x}_i$ occur. Count the l variables occurring in the remaining formula ϕ_R. Let

$$\Sigma = \{\sigma_1, \ldots, \sigma_m\}$$

be a standard enumeration of all possible disjunctive clauses, formed from $x_1, \ldots, x_l$ and their complements, with at most three literals per clause and no variable occurring twice or both complemented and uncomplemented in a clause. Compute

$$\tau_\phi = -\sum_{\sigma_j \in \phi_R} 8^j,$$

where, as below, we use ϵ to denote the relation of an expression *occurring* in another

expression. [Comment: τ_ϕ is the only quantity computed which depends specifically on ϕ_R, rather than just on the number l of variables occurring in ϕ_R.] Compute:

$$f_i^+ = \sum_{x_i \in \sigma_j} 8^j, \qquad i = 1, 2, \ldots, l,$$

$$f_i^- = \sum_{\bar{x}_i \in \sigma_j} 8^j, \qquad i = 1, 2, \ldots, l.$$

Set $n = 2m + l$ and compute c_j, $j = 0, \ldots, n$, as

$$\begin{aligned} c_0 &= 1 \\ c_j &= -\tfrac{1}{2}8^k, \quad j = 2k - 1, \\ c_j &= -8^k, \quad j = 2k, \qquad\qquad j = 1, \ldots, 2m, \\ c_{2m+j} &= \tfrac{1}{2}(f_j^+ - f_j^-), \qquad\qquad j = 1, 2, \ldots, l, \end{aligned}$$

and

$$\tau = \tau_\phi + \sum_{j=0}^{n} c_j + \sum_{i=1}^{l} f_i^-$$

[Comment: At this point, we have in fact obtained a knapsack problem $\sum_{j=0}^{n} c_j\alpha_j = \tau$, $\alpha_j \in \{-1, +1\}$, which is solvable if and only if ϕ is satisfiable; moreover, for any value of $\alpha_j \in \{-1, +1\}$, $|\sum_{j=0}^{n} c_j\alpha_j - \tau| < 8^{m+1}$, so the knapsack problem is equivalent to $\sum_{j=0}^{n} c_j\alpha_j \equiv \tau$ modulo 8^{m+1}, $\alpha_j \in \{-1, +1\}$. These assertions will become clear from the proof of correctness.]

Determine the first $n + 1$ primes, $p_0, \ldots, p_n$, exceeding

$$(4(n + 1)\, 8^{m+1})^{1/(n+1)}.$$

[This in fact never exceeds 12, so we can set $p_0 = 13$.]

Determine parameters θ_j, $j = 0, 1, \ldots, n$, as: the least $\theta_j \in \omega$ such that

$$\theta_j \equiv c_j \text{ modulo } 8^{m+1},$$

$$\theta_j \equiv 0 \text{ modulo } \prod_{i \neq j}^{n} p_i^{n+1},$$

$$\theta_j \not\equiv 0 \text{ modulo } p_j.$$

Compute $H = \sum_{j=0}^{n} \theta_j$, $K = \prod_{j=0}^{n} p_j^{n+1}$ and output:

(a) for Theorem 1:

$$(K + 1)^3 \cdot 2 \cdot 8^{m+1} \cdot (H^2 - x_1^2) + K(x_1^2 - \tau^2) - x_2 \cdot 2 \cdot 8^{m+1} \cdot K = 0,$$

(b) for Theorem 2:

$$x^2 \equiv (2 \cdot 8^{m+1} + K)^{-1} \cdot (K\tau^2 + 2 \cdot 8^{m+1}H^2) \text{ modulo } 2 \cdot 8^{m+1} \cdot K, \qquad 0 \leqslant x \leqslant H,$$

where $(2 \cdot 8^{m+1} + K)^{-1}$ is the inverse of $(2 \cdot 8^{m+1} + K)$ modulo $2 \cdot 8^{m+1} \cdot K^n$.

2.2. *Analysis of Computation Time*

l, and hence m and n, are bounded by a polynomial in the length of the input ϕ. Hence, by the Prime Number Theorem, the primes $p_0, \ldots, p_n$ are also bounded by such a polynomial. It follows that the sizes (numbers of digits in binary representation) of p_j^{n+1}, K, and H are bounded by a polynomial in the length of ϕ; hence the same is true of the output of the algorithm.

Moreover, we can obtain all quantities needed deterministically within polynomial time in the length of the input: The primes can be found as we have exponential time in their length to do so; i.e., we can afford to sieve for the primes. Each θ_j is of the form

$$\lambda_j \cdot \prod_{\substack{i=0 \\ i \neq j}}^{n} p_i^{n+1} \quad \text{or} \quad (\lambda_j + 8^{m+1}) \cdot \prod_{\substack{i=0 \\ i \neq j}}^{n} p_i^{n+1},$$

where

$$\lambda_j \equiv c_j \cdot \left(\prod_{\substack{i=0 \\ i \neq j}}^{n} p_i^{n+1} \right)^{-1} \text{ modulo } 8^{m+1}, \qquad 0 \leqslant \lambda_j < 8^{m+1},$$

and the inverse can be found in polynomial time [10] using the Euclidean algorithm [16]. All other computations are trivially polynomial time, given the bounds on the numbers involved.

2.3. *Proof of Correctness*

In this section, $| x |$ will denote the absolute value of x.

We first show that the original propositional formula ϕ is satisfiable if and only if

$$\sum_{j=0}^{n} \theta_j \alpha_j \equiv \tau \text{ modulo } 8^{m+1}, \qquad \alpha_j \in \{-1, +1\}, \quad j = 0, \ldots, n,$$

is solvable.

Clearly, ϕ_R is satisfiable if and only if ϕ is; ϕ_R is satisfiable if and only if there is a valuation $r: \{x_1, \ldots, x_l\} \to \{0, 1\}$ such that for each disjunctive clause $\sigma_k \in \{\sigma_1, \ldots, \sigma_m\}$

$$0 = R_k \begin{cases} = y_k - \sum_{x_i \in \sigma_k} r(x_i) - \sum_{\bar{x}_i \in \sigma_k} (1 - r(x_i)) + 1, & \text{if } \sigma_k \in \phi_R, \\ = y_k - \sum_{x_i \in \sigma_k} r(x_i) - \sum_{\bar{x}_i \in \sigma_k} (1 - r(x_i)), & \text{if } \sigma_k \notin \phi_R, \end{cases}$$

is solvable by $y_k \in \{0, 1, 2, 3\}$. (In this definition of R_k, ϵ abbreviates "occurs in.") For convenience on a technical point later in the proof, we add an artificial condition

$$0 = R_0 = \alpha_0 + 1, \qquad \alpha_0 \in \{-1, +1\},$$

which does not influence the satisfiability of the system.

For any ϕ_R, any valuation r: $\{x_1,\ldots, x_l\} \to \{0, 1\}$, and any $y_k \in \{0, 1, 2, 3\}$, we have

$$-3 \leqslant R_k \leqslant 4, \qquad k = 1, 2,\ldots, m,$$
$$0 \leqslant R_0 \leqslant 2$$

and therefore

$$R_k = 0, \qquad k = 0, 1,\ldots, m \Leftrightarrow \sum_{k=0}^{m} R_k \cdot 8^k = 0$$

and

$$\left| \sum_{k=0}^{m} R_k \cdot 8^k \right| < 8^{m+1}$$

so that

$$R_k = 0, \qquad k = 0, 1,\ldots, m \Leftrightarrow \sum_{k=0}^{m} R_k 8^k \equiv 0 \text{ modulo } 8^{m+1}.$$

In this condition, we replace the variables y_k and $r(x_i)$ by $\{-1, +1\}$-valued variables α_{2k-1}, α_{2k} and α_{2m+i}, respectively:

$$y_k = \tfrac{1}{2}[(1 - \alpha_{2k-1}) + 2 \cdot (1 - \alpha_{2k})],$$
$$r(x_i) = \tfrac{1}{2}(1 - \alpha_{2m+i}).$$

Let R_k' result from R_k by these substitutions. Then the condition

$$\sum_{k=0}^{m} R_k' 8^k \equiv 0 \text{ modulo } 8^{m+1}$$

contains only the $\{-1, +1\}$-valued variables $\alpha_0,\ldots, \alpha_{2m+l}$, and is solvable if and only if ϕ is satisfiable. By rearrangement of terms, using the quantities defined in the algorithm, the condition can be rewritten as

$$\sum_{j=0}^{n} c_j \alpha_j \equiv \tau \text{ modulo } 8^{m+1}, \qquad \alpha_j \in \{-1, +1\},$$

which by the definition of θ_j, $j = 0,\ldots, n$, is equivalent to

$$\sum_{j=0}^{n} \theta_j \alpha_j \equiv \tau \text{ modulo } 8^{m+1}, \qquad \alpha_j \in \{-1, +1\}.$$

LEMMA 1. *Let K and H be as in the algorithm. The general solution of the system*

$$0 \leqslant |x| \leqslant H, \qquad x \in \mathbb{Z}, \tag{1}$$
$$(H + x)(H - x) \equiv 0 \bmod K \tag{2}$$

is given by

$$x = \sum_{j=0}^{n} \alpha_j \theta_j , \qquad \alpha_j \in \{-1, +1\}, \quad j = 0, 1, \ldots, n.$$

Proof. (of Lemma 1). It is easy to verify that all x of the given form satisfy the system. We now show that these are the only solutions.

Let x be a solution to the system (1)–(2). Then

$$(H + x)(H - x) \equiv 0 \bmod (p_j)^{n+1}, \qquad j = 0, 1, \ldots, n.$$

Assume (for reductio) that for some j_0,

$$p_{j_0} \mid (H + x) \qquad \text{and} \qquad p_{j_0} \mid (H - x).$$

(Notation: $a \mid b$ means a divides b; this is equivalent to $b \equiv 0 \bmod a$.) Then $p_{j_0} \mid (H + x) + (H - x) = 2H$. But $p_{j_0} > 2$, p_{j_0} prime, so we must have $p_{j_0} \mid H$, i.e., $p_{j_0} \mid \sum_j^n \theta_j$. But by definition of θ_j, $p_{j_0} \mid \theta_j$ for all $j \neq j_0$. Hence it would have to be that $p_{j_0} \mid \theta_{j_0}$, contradicting the third condition in the definition of θ_{j_0}.

Thus rejecting our assumption, we conclude that for each j, p_j^{n+1} divides exactly one of $(H + x)$ and $(H - x)$. We define

$$\begin{aligned} \alpha_j &= 1, \qquad \text{if} \quad p_j^{n+1} \mid (H - x), \\ &= -1, \qquad \text{if} \quad p_j^{n+1} \mid (H + x), \\ x' &= \sum \alpha_j \theta_j . \end{aligned}$$

Then we get: (as $x' \equiv \alpha_j \theta_j \equiv \alpha_j H \equiv x$ modulo p_j^{n+1})

$$x' \equiv x \bmod p_j^{n+1} \qquad \text{for all } j \text{ so} \quad x' \equiv x \bmod K,$$

$$\left.\begin{aligned} -H \leqslant x' \leqslant H \\ -H \leqslant x \leqslant H \end{aligned}\right\} \Rightarrow | x - x' | \leqslant 2H.$$

But by our choice of $p_j \geqslant (4(n + 1)\, 8^{m+1})^{1/(n+1)}$, and the fact that

$$\lambda_j = \theta_j \Big/ \prod_{\substack{i=0 \\ i \neq j}}^{n} p_i^{n+1} < 2 \cdot 8^{m+1} \qquad \text{(for each } j\text{)},$$

each term of H is bounded by $K/2(n + 1)$. Hence $2H < K$; we now conclude that $x = x'$. Thus any solution of (1)–(2) is indeed of the form given. (End of proof.)

Using the lemma, we find that the condition

$$\sum_{j=0}^{n} \theta_j \alpha_j \equiv \tau \text{ modulo } 8^{m+1}, \qquad \alpha_j \in \{-1, +1\},$$

is equivalent to the system

$$\begin{aligned} &\text{(i)} \quad 0 \leqslant |x| \leqslant H, \quad x \in \mathbb{Z}, \\ &\text{(ii)} \quad x \equiv \tau \text{ modulo } 8^{m+1}, \\ &\text{(iii)} \quad (H+x)(H-x) \equiv 0 \text{ modulo } K. \end{aligned} \tag{I}$$

LEMMA 2. *Let τ be odd, $x \in \mathbb{Z}$, $k \geqslant 3$.*

$$(\tau + x)(\tau - x) \equiv 0 \bmod 2^{k+1} \Leftrightarrow \text{either } \tau + x \equiv 0 \text{ or } \tau - x \equiv 0 \bmod 2^k.$$

(The straightforward proof is left to the reader.)

We note that in our case the conditions of Lemma 2 are satisfied. Hence the system (I) is satisfiable if and only if system (II) below is satisfiable:

$$\begin{aligned} &\text{(i)} \quad 0 \leqslant |x| \leqslant H, \quad x \in \mathbb{Z}, \\ &\text{(ii)} \quad (\tau + x)(\tau - x) \equiv 0 \bmod 2 \cdot 8^{m+1}, \\ &\text{(iii)} \quad (H+x)(H-x) \equiv 0 \bmod K. \end{aligned} \tag{II}$$

For if x satisfies (I), clearly x satisfies (II); if x satisfies (II), then either $\tau + x \equiv 0 \bmod 8^{m+1}$ or $\tau - x \equiv 0 \bmod 8^{m+1}$; in the second case (I) is satisfied by x, in the first case, $-x$ satisfies conditions (i) and (iii) common to both systems, and

$$-x \equiv \tau \bmod 8^{m+1},$$

so that $-x$ satisfies system (I).

Finally, system (II) is equivalent to

$$\begin{aligned} &\text{(i)} \quad 0 \leqslant x_1 \leqslant H, \quad x_1 \in \mathbb{Z}, \\ &\text{(ii)} \quad \lambda_1 \cdot 2 \cdot 8^{m+1}(H^2 - x_1^2) + \lambda_2 K(\tau^2 - x_1^2) \equiv 0 \text{ modulo } 2 \cdot 8^{m+1} \cdot K \\ &\text{(iii)} \quad \gcd(\lambda_1, K) = \gcd(\lambda_2, 2 \cdot 8^{m+1}) = 1; \ \lambda_1, \lambda_2 \in \mathbb{Z}. \end{aligned} \tag{III}$$

For system (II) only involves x^2 and $|x|$, so that we may assume $x \geqslant 0$ without loss of generality; and the congruences (ii) and (iii) of (II) are equivalent to (ii), (iii) of (III) because $2 \cdot 8^{m+1}$ and K are relatively prime. The parameters λ_1, λ_2 of system (III) can be freely chosen subject to conditions (III)(iii), and we will make different choices for the proofs of Theorems 1 and 2; conditions (III)(i), (ii) are satisfiable either for all λ_1, λ_2 satisfying (III)(iii) or for no such λ_1, λ_2.

We now complete the arguments separately for Theorems 1 and 2.

(a) *Theorem* 1. We choose $\lambda_1 = (K+1)^3$, $\lambda_2 = -1$, clearly satisfying (III)(iii). Now for $x_1 > 0$

$$(K+1)^3 \cdot 2 \cdot 8^{m+1} \cdot (H^2 - x_1^2) + K(x_1^2 - \tau^2) \geqslant 0 \Leftrightarrow 0 \leqslant x_1 \leqslant H.$$

For the first inequality can be written as

$$x_1^2 \leqslant H^2 + \frac{H^2 - \tau^2}{(K+1)^3 \cdot K^{-1} \cdot 2 \cdot 8^{m+1} - 1} = \text{RHS}.$$

We estimate τ by setting

$$B = \sum_{k=1}^{m} 8^k = (8/7)(8^m - 1),$$

and

$$\tau = \tau_\phi + \sum_{j=0}^{2m} c_j + \tfrac{1}{2} \sum_{i=1}^{l} (f_i^+ - f_i^- + 2f_i^-), \qquad -B \leqslant \tau_\phi \leqslant 0,$$

$$\sum_{j=0}^{2m} c_j = 1 - \tfrac{3}{2}B, \qquad B \leqslant \sum_{i=1}^{l} f_i^+ + f_i^- \leqslant 3B$$

so $|\tau| \leqslant 2B - 1 < 8^{m+1} < H$. Therefore RHS satisfies $H^2 < \text{RHS} < H^2 + 1$. It follows that the equation output by the algorithm is solvable by x_1, $x_2 \in \omega$ if and only if system (III) is satisfiable, i.e., if ϕ is satisfiable.

(b) *Theorem* 2. We choose $\lambda_1 = \lambda_2 = 1$, satisfying (III)(iii). Then (III)(ii) becomes

$$(2 \cdot 8^{m+1} + K)\, x^2 \equiv K\tau^2 + 2 \cdot 8^{m+1} H^2 \text{ modulo } 2 \cdot 8^{m+1} \cdot K$$

and as $2 \cdot 8^{m+1} + K$ is relatively prime to $2 \cdot 8^{m+1} \cdot K$, it has an inverse modulo $2 \cdot 8^{m+1} \cdot K$. Multiplying by the inverse, we obtain the congruence condition output by the algorithm. Thus again the conditions output by the algorithm are satisfiable if and only if system (III) is satisfiable.

2.4. *Comments on the Reduction*

(1) It is of broader interest to clarify the basic idea of the reduction algorithm from which a general method for reducing computational problems to Diophantine equations by deterministic computation can be derived. The crucial elements are contained in the system of definitions in the algorithm, and Lemma 1 of the above proof. A version of this pared to bare essentials may be obtained by picking $p_1, \ldots, p_n$ any sufficiently large primes, and for all j, θ_j minimal such that

$$\theta_j \equiv 2^{j-1} \text{ modulo } 2^n,$$

$$\theta_j \equiv 0 \text{ modulo } \prod_{\substack{i=1 \\ i \neq j}}^{n} (p_i)^{n+1},$$

$$\theta_j \not\equiv 0 \text{ modulo } p_j$$

and

$$H = \sum \theta_j , \qquad K = \prod_{i=1}^{n} (p_i)^{n+1}.$$

We then obtain

LEMMA 3 (Conversion Lemma). *For any* $\alpha \in \omega$, *i.e.*, $\alpha = \sum_{i=0}^{n-1} \alpha_i 2^i + \bar{\alpha} 2^n$, $\alpha_i \in \{0, 1\}$, $\bar{\alpha} \in \omega$, *the unique* $a \in \omega$ *satisfying*

(i) $a < K$,

(ii) $a \equiv \alpha \bmod 2^n$,

(iii) $a(a - H) \equiv 0 \bmod K$

is

$$a = \sum_{j=1}^{n} \alpha_{j-1} \theta_j ,$$

α_j *coefficients of binary representation of* α *above.*

The proof of Lemma 3 is of course analogous to that of Lemma 1 above.

The crucial idea in our encoding of the satisfiability problem is to provide a means of going back and forth between the representation of a sequence as the digits of a number in some base b (e.g., 2), and as the residues of a number with respect to a system of relatively prime moduli. The first allows a global description of the shifting of the sequence, the second allows global formulation (i.e., for the whole sequence in a single congruence) of a condition on the individual elements of the sequence. All known reductions of recognition of correct Turing machine computation of Diophantine equations with a number of variables independent of input size provide some means of reconciling these very different kinds of operations on the sequence studied; doing so is the fundamental problem of such reductions and the principal challenge in obtaining tight bounds on the complexity of Diophantine decision problems.

(2) If $S \in NP \cap NP^c$, i.e., $S \in NP$ and $S^c \in NP$, then the reduction algorithm described in the proof of Theorem 1 reduces the questions

$$x \in S ?$$
$$x \in S^c ?$$

to the question of the solvability of two very closely related Diophantine equations. This fact might be of help in studying the class $NP \cap NP^c$. As an example, we suggest the

OPEN PROBLEM 1. Is there a complete set in $NP \cap NP^c$ (i.e., a set in $NP \cap NP^c$ to which every set in $NP \cap NP^c$ can be reduced by a deterministic polynomial time algorithm)?

As a candidate we suggest the set S

$$S = \{\langle \alpha, \beta \rangle : \alpha, \beta \in \omega; \exists z \in \omega, 1 < z < \beta \text{ and } z \mid \alpha\}.$$

S is in $NP \cap NP^c$ and the problem of accepting S is computationally deterministic polynomial time equivalent to prime factorization of numbers, and hence probably not in P (see [14] for these assertions).

3. Nondeterministic Diophantine Machines

We introduce a class of nondeterministic Turing machines with restricted, purely numerical, computational ability: Given a multivariable polynomial $p(x_1, \ldots, x_m, y_1, \ldots, y_n)$ with integer coefficients, the corresponding *nondeterministic Diophantine machine* (NDDM) is a nondeterministic Turing machine with the following algorithm:

"On input $a_1, \ldots, a_m \in \omega$, guess $b_1, \ldots, b_n \in \omega$. If $p(a_1, \ldots, a_m, b_1, \ldots, b_n) = 0$ then accept $\langle a_1, \ldots, a_m \rangle$; otherwise halt without accepting."

For example, if $p(x_1, y_1) = x_1 - y_1^2$, then the corresponding NDDM has the algorithm

"On input $a_1 \in \omega$; guess $b_1 \in \omega$. If $a_1 - b_1^2 = 0$, accept a_1."

It is easy to see that this NDDM accepts exactly the set of perfect squares, and in polynomial time.

It is of interest to model computation by NDDM's for two reasons. To the extent that NDDM's are equally capable as arbitrary nondeterministic Turing machines, it will be advantageous to study NDDM's rather than arbitrary machines because the directly number-theoretical description of sets accepted by NDDM's facilitates application of techniques and results from number theory to the study of computation. On the other hand, NDDM's by their definition isolate "arithmetical" from "combinatorial" aspects of nondeterministic computation; studying the extent to which the computational abilities of NDTM's exceed those of NDDM's (if at all) will clarify the importance of having resources for combinatorial rather than numerical computation in a nondeterministic setting.

The development of the theory of Diophantine definability in number theory, in connection with Hilbert's 10th problem, has made it possible to compare the models; for the Diophantine relations are exactly those accepted by an NDDM. Thus we have as a corollary to Matijasevic's theorem mentioned in the introduction, that the relations accepted by NDDM's are exactly the relations accepted by Turing machines.

Our question can now be sharpened: How does the efficiency of NDDM's compare with that of NDTM's? We show in [1, 3] that the restriction to the numerical operations of NDDM's causes at most an exponential loss in efficiency: A relation accepted in time t on a NDTM is accepted in at most time 2^{10t^2} on a NDDM (for a large class of arbitrarily large or small running time functions t). No lower bounds on the loss of efficiency are known; we conjecture that essentially no efficiency is lost. This is supported by evidence in [1, 3].

For close comparison of the efficiency of NDDM's and NDTM's we compare the class D of relations accepted by NDDM's in polynomial time to NP and P. The following definitions are useful:

(i) (alternative characterization of D): For all $n \in \omega$, D^n is the set of all numerical relations R definable by a formula of the form:

$$\langle x_1, x_2, \ldots, x_m \rangle \in R \Leftrightarrow \exists y_1, \ldots, y_n \leqslant 2^{q(|x_1+x_2+\cdots+x_m|)}[P(x_1, x_2, \ldots, x_m, y_1, \ldots, y_n) = 0],$$

where q and P are polynomials, and, as throughout this section, $| x |$ denotes the length of x in binary.

Then we have:

$$D = \bigcup_{i \in \omega} D^i.$$

(ii) For any m-ary numerical relation R and any l-ary numerical relation S: *R is D-reducible to S* (notation: $R \leqslant_D S$) if and only if R is definable by a formula of the form:

$$\langle x_1, \ldots, x_m \rangle \in R \Leftrightarrow \exists y_1, \ldots, y_l, y_{l+1}, \ldots, y_n \leqslant 2^{q(|x_1+\cdots+x_m|)}[P(x_1, \ldots, x_m, y_1, \ldots, y_n) = 0 \,\&\, \langle y_1, \ldots, y_l \rangle \in S],$$

where q and P are polynomials.

(iii) For any numerical relation R in NP: *R is D-complete* if and only if every other numerical relation in NP is D-reducible to R. Clearly if R is D-complete then $R \in D \Leftrightarrow NP = D$.

We can now use the results of the previous section to show that all nondeterministic polynomial time computation can be decomposed into deterministic polynomial time computation and polynomial time computation on an NDDM, by a fixed deterministic Turing machine and an essentially fixed NDDM. (Another such decomposition, perhaps more natural, but requiring more formidable methods of proof, is given in [3].)

THEOREM 3. (a) *There is an NP-complete problem in D (in fact even in D^2).*

(b) *There is a D-complete problem in P.*

COROLLARY. (a) $D \subseteq P \Rightarrow P = NP$,

(b) $P \subseteq D \Rightarrow D = NP$.

Proof. (a) Let $p(x_1, x_2, x_3, y_1, y_2) = x_1 y_1^2 + x_2 y_2 - x_3$. Then the NDDM corresponding to $p(x_1x_2x_3, y_1y_2)$ accepts the relation S:

$$S = \{\langle \alpha, \beta, \gamma \rangle : \alpha, \beta, \gamma \in \omega, \exists y_1 y_2 \in \omega[\alpha y_1^2 + \beta y_2 - \gamma = 0]\},$$

which is NP-complete as it is in deterministic polynomial time 1–1 correspondence with the set of Diophantine equations asserted to be NP-complete in Theorem 1; and the NDDM runs in polynomial time.

(b) We must find a relation $S \in P$ and for any NDTM M accepting a set in NP, a D-reducing Diopantine equation $p_M(\cdots) = 0$. We consider a nondeterministic simulation of an arbitrary NDTM M (where $M \in \omega$ will serve as the index of the NDTM)

which will accept in nondeterministic polynomial time if M does. The computation involved in this simulation will be "divided" into a deterministic part and a nondeterministic part; the deterministic part will give us the definition of a relation S which will be in P; the nondeterministic part will give us the definition of the "reducing" NDDM, for it will be a Diophantine relation.

The Simulation Algorithm. "For NDTM M, on input $x \in \omega$: Guess a time $t \in \omega$ (such that M on input x may halt within t steps). By Cook's algorithm [6], compute a propositional formula in disjunctive form with at most 3 literals per clause, satisfiable if and only if M on input x halts within t steps. By the reduction algorithm in the proof of Theorem 1, compute the appropriate $\alpha, \beta, \gamma \in \omega$. Guess x_1, x_2 (such that $\alpha x_1^2 + \beta x_2 - \gamma = 0$ may hold). If $\alpha x_1^2 + \beta x_2 - \gamma = 0$, accept x. Halt."

$$S = \{\langle M, x, t, \alpha, \beta, \gamma\rangle : M, x, t, \alpha, \beta, \gamma \in \omega, \text{ and } \alpha, \beta, \gamma \text{ are as computed in the simulation algorithm from } M, x, t\}$$

$$p_M(x_1, y_1, \ldots, y_8) = (y_4 y_7^2 + y_5 y_8 - y_6)^2 + (y_1 - M)^2 + (y_2 - x_1)^2.$$

Note that

$$p_M(\cdots) = 0 \Leftrightarrow y_4 y_7^2 + y_5 y_8 - y_6 = 0,$$
$$y_1 = M,$$
$$y_2 = x_1.$$

Now assume that M is a NDTM accepting a set $S_M \in NP$, i.e., M has a running-time function which is a polynomial in the length of the input. Then the simulation algorithm will accept S_M in polynomial time and needs only to guess numbers bounded by

$$2^{q_M(|x|)}$$

for some polynomial $q_M(\cdot)$ which could be determined from the running time of M by an analysis of the reduction algorithm used in proving Theorem 1 above. Thus we have, for all $x_1 \in \omega$:

$$x_1 \in S_M \Leftrightarrow \exists y_1, \ldots, y_8 \in \omega; \qquad y_1, \ldots, y_8 \leqslant 2^{q_M(|x_1|)}$$

such that

$$p_M(x_1, y_1, \ldots, y_8) = 0,$$
$$\langle y_1, \ldots, y_6\rangle \in S,$$

so that M is indeed D-reducible (by p_M) to $S \in P$. As M was arbitrary and S does not depend on M, S is D-complete. (End of proof.)

Theorem 3 answers an open problem in [2]: "Find a set $A \in D$ such that for all $B \subseteq \omega$, if $B \in D$, then B is polynomial reducible to A." That any such set would be *NP*-complete (as Theorem 3 implies) was rather unexpected. One would have expected the D^i, $i \in \omega$ to be a hierarchy of progressively harder problems (in the sense of P-reducibility);

moreover, it was suspected that number-theoretic problems obviously in NP would be less than NP-complete: The deep structure of number theory should allow development of nontrivial and efficient algorithms for such problems. Theorem 3 indicates that this is wrong.

There are various natural subdivisions of the class D corresponding to characteristics of the defining polynomials of NDDM's computing sets in D: number of variables, degree in the variables, and the magnitude of the time bound. These suggest many problems of classification of sets and may prove significant for complexity theory. We discuss several of the possibilities.

$D^{2(K)}$ will denote the subclass of D^2, where the relevant polynomial $P(x_1, \ldots, x_m, y_1, y_2)$ is of degree $\leqslant K$ in the variables y_1, y_2. Clearly, the set S in the proof of Theorem 3(a) is in $D^{2(2)}$. It then easily follows from Theorem 3(a) that

THEOREM 4. *The following are equivalent.*

(a) $\bar{S} \in NP$,

(b) $(D^{2(2)})^c \subseteq NP$,

(c) $(NP)^c = NP$,

where A^c denotes the set of complements of sets in A and $\bar{A}$ denotes the complement of A.

Theorem 4 is, in many respects, as strong as possible: We can show [3] that

(i) $(D^{2(1)})^c \subseteq P \subseteq NP$,

(ii) $(D^1)^c \subseteq D \cap P \subseteq NP$.

It is very interesting to consider the extent of $D^{2(2)}$.

Consider the following sets:

$$S_1 = \{\alpha \mid \exists x, y \in \omega: y^2 - \alpha x^2 = 1\},$$

$$S_2 = \{\alpha \mid \alpha \text{ composite}\},$$

$$S_3 = \{\langle \alpha, \beta \rangle \mid \exists z \in \omega: 1 < z < \beta \text{ and } z \text{ divides } \alpha\},$$

$$S_4 = \{\langle \alpha, \beta, \gamma \rangle \mid \exists y, z \in \omega: \alpha y^2 + \beta z - \gamma = 0\}.$$

All of these are in $D^{2(2)}$. But also:

S_1 is in P (see [16]).

S_2 is in P, if the Extended Riemann Hypothesis is true (see [14, 15]).

S_3 : As noted at the end of Section 2, $S_3 \in NP \cap NP^c$; probably $S_3 \notin P$.

S_4 is NP-complete.

These examples illustrate that $D^{2(2)}$ is a microcosm of the principal subclasses of NP. This suggests that a more detailed determination of the extent of $D^{2(2)}$ would be valuable; for example:

OPEN PROBLEM 2. (a) Show that $D^{2(2)} \neq NP$.

(b) Show that $P \not\subseteq D^{2(2)}$.

(Obviously, the second implies the first.)

(c) Does every degree in NP with respect to $\leqslant_p$ (deterministic polynomial time reducibility) have a representative in $D^{2(2)}$?

The single most significant open problem about NDDM's is whether NDDM's are in fact equivalent to NDTM's at the polynomial time level, i.e.,

OPEN PROBLEM 3. Is $D = NP$?

Especially if the answer to problem 3 is affirmative, it will be important to investigate natural subdivisions of D with respect to their possible computational significance, i.e., in a careful analysis of the effect of resource bounds. Thus one would like to know:

OPEN PROBLEM 4 (role of number of variables). Are the inclusions $D^i \subseteq D^{i+1}$ strict?

OPEN PROBLEM 5 (role of time bounds). For $k \geqslant 1$, let $D(k)$ be the set of all numerical relations definable by a formula of the form

$$\langle x_1, \ldots, x_m \rangle \in R \Leftrightarrow \exists y_1 \cdots y_n \leqslant 2^{c(|x_1+\cdots+x_m|)^k} : P(x_1 \cdots x_m, y_1 \cdots y_n) = 0,$$

where $c > 0$ and P is a polynomial. If $k > l$, then $D(l) \subseteq D(k)$; $D(1)$ is just the class where the definition can be chosen as

$$\exists y_1 \cdots y_n \leqslant q(x_1 \cdots x_m) : P(x_1 \cdots x_m, y_1 \cdots y_n) = 0,$$

P, q polynomials.

(a) The relation $x = y^z$ is in $D(2)$ [2]. Is it in $D(1)$?

(b) Are the inclusions $D(l) \subseteq D(k)$, for $k > l$, strict? An affirmative answer would follow from $D = NP$, by use of diagonalization over nondeterministic Turing machines running in time n^k. But this question could be independent of $NP = D$.

If, on the other hand, $D \neq NP$, then D and P would be analogous as subclasses of NP determined by qualitative restrictions on the nature of computation allowed. In this case it would be of interest to raise the kinds of questions concerning D as are presently raised concerning deterministic polynomial-time computation: Complexity classification of problems in NP with respect to D-reduction.

OPEN PROBLEM 6. What is the structure of NP under $\leqslant_D$ (D-reducibility)? Are there D-(reducibility) degrees between 0 (the degree of sets in D) and the degree of the D-complete set of Theorem 3?

OPEN PROBLEM 7 (classification of the set *Pr* of prime numbers). $Pr \in NP$ [17], and $Pr \in P(ERH)$ [15].

(a) Is $Pr \in D$?

(b) Is *Pr* D-complete?

ACKNOWLEDGMENT

The authors thank Julia Robinson and Bill Sakoda for comments on an earlier draft of this article.

REFERENCES

1. L. ADLEMAN, "Number Theoretic Aspects of Computational Complexity," Ph.D. Dissertation, University of California, Berkeley, 1976.
2. L. ADLEMAN AND K. MANDERS, The computational complexity of decision procedures for polynomials, *in* "16th Annual IEEE Symposium on Foundations of Computer Science," pp. 169–177, IEEE, New York, 1975.
3. L. ADLEMAN AND K. MANDERS, Diophantine complexity, *in* "17th Annual IEEE Symposium on Foundations of Computer Science," pp. 81–88, IEEE, New York, 1976.
4. A. BAKER, Contributions to the theory of Diophantine equations. I. On the representation of integers by binary forms, *Philos. Trans. Roy. Soc. London Ser. A* **263** (1967/68), 173–191.
5. A. BAKER, "Transcendental Number Theory," Cambridge Univ. Press, 1975.
6. S. A. COOK, The complexity of theorem-proving procedures, *in* "Conf. Rec. 3rd ACM Symposium on Theory of Computing," pp. 151–158, 1971.
7. G. HARDY AND E. WRIGHT, "An Introduction to the Theory of Numbers," Oxford Univ. Press (Clarendon), London, 1938.
8. D. HILBERT, Mathematische Probleme: Vortrag gehalten auf dem internationalen Mathematiker-Kongress zu Paris, 1900, *Nachr. Akad. Wiss. Göttingen, Math.-Phys. Kl. II* (1900), 253–297.
9. R. M. KARP, Reducibility among combinatorial problems, *in* "Complexity of Computer Computation" (R. N. Miller and J. W. Thatcher, Eds.), pp. 85–104, Plenum, New York, 1972.
10. M. LAMÉ, Note sur la limite du nombre des divisions dans la recherche du plus grand commun diviseur . . ., *C. R. Acad. Sci. Paris Sér. A-B* **19** (1844), 867–869.
11. Y. MATIJASEVIC, Enumerable sets are Diophantine, *Dokl. Akad. Nauk SSSR* **191** (1970), 279–282, (Russian).
12. Y. MATIJASEVIC AND J. ROBINSON, Reduction of an arbitrary Diophantine equation to one in 13 unknowns, *Acta Arith.* **27** (1975), 521–553.
13. Y. MATIJASEVIC, private communication.
14. G. L. MILLER, Ph.D. Dissertation, University of California, Berkeley, 1975.
15. G. L. MILLER, Riemann's hypothesis and tests for primality, *J. Comput. System Sci.* **13** (1976), 300–317.
16. I. NIVEN AND H. ZUCKERMAN, "An Introduction to the Theory of Numbers," Wiley, New York, 1972.
17. V. PRATT, Every prime has a succinet certificate, *SIAM J. Computing* **4** (1975), 214–220.
18. A. THUE, Über Annäherungswerte algebraischer Zahlen, *J. Reine Angew. Math.* **135** (1909), 284–305.