THE JOURNAL OF SYMBOLIC LOGIC
Volume 49, Number 3, Sept. 1984

REGISTER MACHINE PROOF OF THE THEOREM ON EXPONENTIAL DIOPHANTINE REPRESENTATION OF ENUMERABLE SETS

J. P. JONES AND Y. V. MATIJASEVIČ[1]

§1. Introduction. The purpose of the present paper is to give a new, simple proof of the theorem of M. Davis, H. Putnam and J. Robinson [1961], which states that every recursively enumerable relation $A(a_1,\ldots,a_n)$ is *exponential diophantine*, i.e. can be represented in the form

$$(1)\quad A(a_1,\ldots,a_n) \leftrightarrow \exists x_1,\ldots,x_m[R(a_1,\ldots,a_n,x_1,\ldots,x_m) = S(a_1,\ldots,a_n, x_1,\ldots, x_m)],$$

where $a_1,\ldots,a_n$, $x_1,\ldots,x_m$ range over natural numbers and R and S are functions built up from these variables and natural number constants by the operations of addition, $A + B$, multiplication, AB, and exponentiation, A^B. We refer to the variables $a_1,\ldots,a_n$ as *parameters* and the variables $x_1,\ldots,x_m$ as *unknowns.*

Historically, the Davis, Putnam and Robinson theorem was one of the important steps in the eventual solution of Hilbert's tenth problem by the second author [1970], who proved that the exponential relation, $a = b^c$, is diophantine, and hence that the right side of (1) can be replaced by a polynomial equation. But this part will not be reproved here. Readers wishing to read about the proof of that are directed to the papers of Y. Matijasevič [1971a], M. Davis [1973], Y. Matijasevič and J. Robinson [1975] or C. Smoryński [1972]. We concern ourselves here for the most part only with exponential diophantine equations until §5 where we mention a few consequences for the class NP of sets computable in nondeterministic polynomial time.

The original [1961] proof of the Davis, Putnam and Robinson theorem took as its starting point the theorem of M. Davis [1953] that every recursively enumerable

Received September 10, 1981; revised October 28, 1981 and January 3, 1983.
1980 *Mathematics Subject Classification.* Primary 03D10, 03D25, 10N05.

[1] This new proof was worked out in the Steklov Mathematical Institute, Leningrad, and in the University of Calgary, Department of Mathematics and Statistics, in connection with the programs of scientific exchange of the National Academy of Sciences of the U.S.A., the Academy of Sciences of the U.S.S.R. and the Queens-Steklov Mathematical Exchange Program between Canada and the U.S.S.R. The authors wish to acknowledge also the support of the Natural Sciences and Engineering Research Council of Canada and the valuable advice of Professors Julia B. Robinson and Raphael M. Robinson of the University of California, Berkeley.

0022-4812/84/4903-0009/$02.20

relation could be represented in the form

$$(2) \qquad A(a_1,\ldots,a_n) \leftrightarrow \exists y(\forall z \leq y)\exists x_1,\ldots,x_m[P(a_1,\ldots,a_n,y,z,x_1,\ldots,x_m) = 0].$$

Here a bounded universal quantifier $\forall$ appears. It was then necessary to prove what came to be called the Bounded Quantifier Theorem, namely that you can replace the right side of (2) by an exponential diophantine equation such as appears in (1). The original proof was very complicated and underwent a series of simplifications. The various versions of the Bounded Quantifier Theorem can be found in Davis, Putnam and Robinson [1961], J. Robinson [1969], M. Davis [1973], Y. Matijasevič [1971c], [1972] and [1974], Davis, Matijasevič and Robinson [1976], Hirose and Iida [1973], and Jones [1978]. But we shall not need any of these here. Our new proof avoids completely the Bounded Quantifier Theorem. Also there is no need to establish first the Davis representation (2).

The new proof is based on *register machines* (also called *counter machines* or *program machines*; see M. Minsky [1961], [1967], Z. A. Melzak [1961], J. Lambek [1961], and J. C. Shepherdson and H. E. Sturgis [1963]). A register machine is similar to a Turing machine. The main difference is that the Turing machine uses an infinite tape for storage whereas the register machine uses a finite number of registers, each containing a nonnegative integer. The registers are addressable, and the machine's instructions permit arithmetical operations to be performed on them. Of course the Turing and the register machine are equivalent. The same class of (partial recursive) functions is computable. However, using register machines we avoid the need to arithmetize words over an alphabet (necessary in Matijasevič [1976]). Many mathematicians and logicians interested in the theory of algorithms have also come to the conclusion that the register machine provides a more natural, less awkward model of computability (cf. e.g. E. Börger [1975]).

Since the new proof presupposes very little knowledge of number theory, it would be possible to use it in a course in computer science. It can also be used as the main link in a proof that every Turing computable function is general recursive. The proof that every Turing computable function is register machine computable is very short (Minsky [1967, pp. 170–172, 204–206]).

After the theorem of Davis, Putnam and Robinson was proved in 1961, improvements were made in it. In 1974 the theorem was strengthened to *singlefold unary* exponential diophantine representation. *Singlefold* (Russian *odnokratnoe*; French *univoque*) means that there is at most one m-tuple, $x_1,\ldots,x_m$, satisfying the equation. *Unary* (Russian, *oonarnoe*) means that the exponential equation can be built up by addition and multiplication using only one place exponentials, 2^x, rather than two place exponentials y^x. (The original 1961 paper already established the possibility of unary representation but not singlefold.)

In 1979 it was proved that every r.e. set has a *singlefold* exponential diophantine representation in *three unknowns* (Matijasevič [1979]). In 1982 this was further improved to a *singlefold three-unknown unary* representation (Jones and Matijasevič [1982]).

The new register machine proof, given here in this paper, establishes singlefold unary representation but not in three unknowns.

§2. Exponential diophantine sets. In this section we explain some notation and develop the necessary (minimal) number theory. All variables range over natural numbers (nonnegative integers). q pow 2 means that q is a power of 2. $\lfloor x \rfloor$ and $\lceil x \rceil$ are the floor and the ceiling of x. rem(x, y) = remainder after x is divided by y. A diophantine equation is a polynomial equation in which we are interested only in integer solutions, or, as in this paper, nonnegative integer solutions.

Usually one begins with a diophantine equation and one asks about the set of solutions. However, here in this subject, as Martin Davis has remarked, the usual procedure is turned around. We begin with an interesting set or relation and ask for a diophantine equation defining it (in the sense of (1)). If so, the relation is called *diophantine.* If exponential functions appear, the relation is said to be *exponential diophantine.* A simple example of a diophantine relation is the ordinary ordering relation, $<$, on the natural numbers.

(3) $$a < b \leftrightarrow \exists x[a + x + 1 = b].$$

Since x is unique in (3), we say that $<$ is a *singlefold* diophantine relation. Further examples would be the relations $\leqq$ (less than or equal to), $\neq$ and $|$ (divisibility).

Another example would be the relation of congruence

(4) $$a \equiv b\,(\mathrm{mod}\, c) \leftrightarrow (\exists x)\,[a = b + cx \text{ or } b = a + cx].$$

A disjunction or conjunction of equations (in real numbers) can be combined into an equivalent single equation using

(5) $$A = 0 \text{ or } B = 0 \;\leftrightarrow\; AB = 0, \qquad A = 0 \,\&\, B = 0 \;\leftrightarrow\; A^2 + B^2 = 0.$$

These principles (5) can be used to show that the remainder function is (singlefold) diophantine. For $0 < c$

(6) $$a = \mathrm{rem}(b, c) \leftrightarrow b \equiv a\,(\mathrm{mod}\, c) \quad \text{and} \quad a < c.$$

Next we show that the binomial coefficient relation, $m = \binom{n}{k}$, is exponential diophantine. This was first proved by Julia Robinson [1952]. We give here a modification of her original proof, found later by the second author [1971b]. Both proofs use the fact that the binomial coefficients $\binom{n}{i}$ are *defined* by the binomial theorem,

(7) $$(u + 1)^n = \sum_{i=0}^{n} \binom{n}{i} u^i$$

and the binomial coefficients are just the *digits* in the base u expansion of the number $(1 + u)^n$, when u is sufficiently large, say $u > 2^n \geq \binom{n}{i}$. The *digits* of a number are unique. Hence, $m = \binom{n}{k}$ holds if and only if there exist (unique) u, w and v such that

(8) $$u = 2^n + 1, \quad (u + 1)^n = wu^{k+1} + mu^k + v, \quad v < u^k \quad \text{and} \quad m < u.$$

This proves that the relation $m = \binom{n}{k}$ is singlefold exponential diophantine. The conditions (8) are not unary exponential diophantine because they contain two-place exponential functions. However the two-place exponentials can be replaced by one-place exponentials using the formula

(9) $$\mathrm{rem}(2^{xy^2}, 2^{xy} - x) = x^y \qquad (1 < y).$$

This idea is due essentially to Julia Robinson. Formula (9) is easy to prove using $2^{xy} \equiv x (\operatorname{mod} 2^{xy} - x)$.

Next we define a new binary relation, $\preccurlyeq$ (masking). Suppose that the numbers r and s are written in binary (base 2) notation,

$$r = \sum_{i=0}^{n} r_i 2^i \quad (0 \le r_i \le 1), \qquad s = \sum_{i=0}^{n} s_i 2^i \quad (0 \le s_i \le 1).$$

DEFINITION. *$r \preccurlyeq s$ if and only if each binary digit of r is less than or equal to the corresponding binary digit of s, i.e., for all i, $r_i \le s_i$.*

The relation $\preccurlyeq$ has many interesting properties. It is a partial ordering, and $r \preccurlyeq s$ implies $r \le s$. It is closely related to the operation of taking the so-called *logical and* of two binary numbers

$$\text{(10)} \qquad a \preccurlyeq b \leftrightarrow a \,\&\, b = a,$$

$$\text{(11)} \qquad a \,\&\, b = c \leftrightarrow c \preccurlyeq b \text{ and } b \preccurlyeq a + b - c.$$

One can also use $\preccurlyeq$ to define the set of powers of 2:

$$\text{(12)} \qquad a \operatorname{pow} 2 \leftrightarrow a \preccurlyeq 2a - 1.$$

Also, two $\preccurlyeq$ conditions can always be combined into a single $\preccurlyeq$ condition. If Q pow 2 and $a < Q$ and $b < Q$, then

$$\text{(13)} \qquad a \preccurlyeq b \text{ and } c \preccurlyeq d \leftrightarrow a + cQ \preccurlyeq b + dQ.$$

We need to prove that the relation $\preccurlyeq$ is (singlefold unary) exponential diophantine. The following lemma proves this.

LEMMA. *$r \preccurlyeq s$ if and only if $\binom{s}{r} \equiv 1 \pmod 2$.*

PROOF. This is clear from E. Lucas' theorem [1878a], [1878b]

$$\text{(14)} \qquad \binom{s}{r} \equiv \binom{s_n}{r_n} \cdots \binom{s_1}{r_1}\binom{s_0}{r_0} \pmod p,$$

where r_i and s_i are the p-ary digits of r and s. Here it is necessary that p be a prime. We take $p = 2$.

Lucas' theorem is usually proved by considering the coefficient of the term x^r in the product $(1 + x)^s$. Use the congruence $(1 + x)^{p^i} \equiv 1 + x^{p^i} \pmod p$ and work in the ring $Z_p[x]$ of polynomials in x with coefficients in the field Z_p (cf. e.g. Fine [1947]). For another, different proof of Lucas' theorem, see M. Hausner [1983].

§3. Arithmetization of register machines. The Davis, Putnam and Robinson theorem is usually stated in the form *every recursively enumerable set is exponential diophantine.* (The converse is obviously true. Every exponential diophantine set is recursively enumerable.) For the concept of a *recursively enumerable* set (r.e. set, listable set), there are many equivalent definitions from which to choose. We take here a definition formulated in terms of *register machines.* For additional information about *recursively enumerable sets, recursive sets, recursive* (*computable*) *functions*, and *partial recursive functions*, see the expository books and papers Davis [1958], [1973], [1974], Minsky [1967], Robinson [1969], or Jones [1974].

A *register machine* (Minsky [1961], [1967], Melzak [1961], Lambek [1961], Shepherdson and Sturgis [1963]) has a finite number of separately addressable registers $R1, R2, \ldots, Rr$ and a program consisting of a finite list of commands considered to be written on lines labelled $L0, L1, \ldots, Ll$. The commands are normally executed in the sequence given, but the register machine can be instructed to transfer to a different location. Each register contains a nonnegative integer. Unlike physically existing digital computers, there is no preassigned upper bound on the size of the numbers storable in a register. One also supposes that the machine is capable of inspecting the contents of any register and adding or subtracting 1 provided that this does not produce a negative number in a register. In addition, to a STOP command we suppose that there are five (active) instructions:

	COMMAND	MEANING
(15)	Li GO TO Lk	Proceed to the instruction at location Lk.
(16)	Li IF $Rj < Rm$, GO TO Lk	If the contents of Rj is less than the contents of Rm, then go to Lk. Else go to the next instruction.
(17)	Li $Rj \leftarrow Rj + 1$	Add 1 to the contents of Rj.
(18)	Li $Rj \leftarrow Rj - 1$	Subtract 1 from the contents of Rj.

We permit $\leq$ to occur in the conditional transfer command (16), i.e. we permit also the command

(19) $$Li \quad \text{IF } Rj \leq Rm, \text{ GO TO } Lk.$$

We permit also occurrence of numbers, 0 and 1, in place of names of registers in commands (16) and (19). So, for example, we can simulate the (double) conditional transfer command of Minsky [1967].

(20) $$Li \qquad \text{IF } Rj = 0, \text{ GO TO } Lk,$$

(21) $$Li + 1 \qquad \text{ELSE } Rj \leftarrow Rj - 1.$$

Minsky permitted the subtraction command to occur only in this way (with (21) immediately after (20)), to avoid the problem of an attempted subtraction from an already zero register. We can make here a similar assumption, or we can simply suppose that the program was written in such a way that subtraction of 1 from a zero register never occurs.

The following is an example of a register machine program.

EXAMPLE 1.

$$\begin{array}{ll}
L0 & R2 \leftarrow R2 + 1, \\
L1 & R2 \leftarrow R2 + 1, \\
L2 & \text{IF } R3 = 0, \text{ GO TO } L5, \\
L3 & R3 \leftarrow R3 - 1, \\
L4 & \text{GO TO } L2, \\
L5 & R3 \leftarrow R3 + 1,\ R4 \leftarrow R4 + 1,\ R2 \leftarrow R2 - 1, \\
L6 & \text{IF } 0 < R2, \text{ GO TO } L5,
\end{array}$$

$L7$ $R2 \leftarrow R2 + 1$, $R4 \leftarrow R4 - 1$,
$L8$ IF $0 < R4$, GO TO $L7$,
$L9$ IF $R3 < R1$, GO TO $L5$,
$L10$ IF $R1 < R3$, GO TO $L1$,
$L11$ IF $R2 < R1$, GO TO $L10$,
$L12$ $R1 \leftarrow R1 - 1$, $R2 \leftarrow R2 - 1$, $R3 \leftarrow R3 - 1$,
$L13$ IF $0 < R1$, GO TO $L12$,
$L14$ STOP.

Here there are $r = 4$ registers and $l = 14$ lines in the program. To shorten the program, lines 5, 7 and 12 were written in parallel. This makes no essential difference provided we do not attempt to parallel the same register twice on one line. (Combining $L0$ and $L1$ into one line of code would require changes in (24), (38) and (39)). We do not try to parallelize transfer commands.

To define the concept of a *recursively enumerable set* we take the convention that a register machine M *accepts* a number x, if M, started at $L0$, with x in $R1$ and zero in the other registers, eventually stops, with zero in all the registers.

Minsky [1967] proved that with commands of type (15), (17), (20) and (21), already three registers are sufficient for any computation and for acceptance of r.e. sets. Minsky also proved that two registers are sufficient when one uses a different input format convention (for example 2^x in $R1$ instead of x). (J. M. Barzdin' [1963] proved that some such special input format is necessary with two registers.) Minsky proved also that with additional commands and special input format encoding, one register is sufficient. Here we allow any finite number r of registers.

We can use digits of numbers, written to a base Q, to describe the work of a register machine. Let s be the number of *steps* in the computation (assuming the machine stops). Let $r_{j,t}$ be the *contents* of register Rj at time t ($t = 0, 1, \ldots$). Let $l_{j,t}$ be 1 or 0 according as at time t we do or do not execute the instructions at *location* j (i.e., on line number j in the program). Let

$$R_j = \sum_{t=0}^{s} r_{j,t} Q^t \qquad (0 \le r_{j,t} < Q/2), \tag{22}$$

$$L_i = \sum_{t=0}^{s} l_{i,t} Q^t \qquad (0 \le l_{i,t} \le 1). \tag{23}$$

As a base Q we can use any sufficiently large power of 2. The contents of any register at time t cannot exceed $x + t$. Hence if there are s steps in the program, then $r_{j,t} \le x + s$. So any number Q satisfying

$$x + s < Q/2, \tag{24}$$

$$l + 1 < Q, \tag{25}$$

$$Q \text{ pow } 2 \tag{26}$$

will be large enough. For example we can put $Q = 2^{x+s+l}$ (for singlefoldness).

It will be convenient to have a number I which when written to the base Q has all its digits equal to 1. We can obtain such a number I from the geometric series. If

$$1 + (Q - 1)I = Q^{s+1}, \tag{27}$$

then

$$I = \sum_{t=0}^{s} Q^t. \tag{28}$$

Suppose we start the machine of Example 1 with 2 in $R1$. Then the machine will be seen to halt after $s = 18$ steps with 0 in all the registers. (The set of accepted numbers is the set of primes.)

The work of the machine and the generated numbers $R_1, \dots, R_4$ and $L_0, \dots, L_{14}$ may be visualized as follows, time proceeding leftwards, so the numbers $R_1, \dots, R_4$, $L_0, \dots, L_{14}$ are presented as normally written, in base Q notation,

0 0 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	$R1$	REGISTER
0 0 1 1 2 2 2 2 2 1 1 0 0 1 1 2 2 1 0	$R2$	REGISTER
0 0 1 1 2 2 2 2 2 2 2 2 2 1 1 0 0 0 0	$R3$	REGISTER
0 0 0 0 0 0 0 0 0 1 1 2 2 1 1 0 0 0 0	$R4$	REGISTER
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1	$L0$	$R2 \leftarrow R2+1$,
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0	$L1$	$R2 \leftarrow R2+1$,
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0	$L2$	IF $R3=0$, GO TO $L5$,
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	$L3$	$R3 \leftarrow R3-1$,
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	$L4$	GO TO $L2$,
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0	$L5$	$R3 \leftarrow R3+1$, $R4 \leftarrow R4+1$, $R2 \leftarrow R2-1$,
0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0	$L6$	IF $0<R2$, GO TO $L5$,
0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0	$L7$	$R2 \leftarrow R2+1$, $R4 \leftarrow R4-1$,
0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0	$L8$	IF $0<R4$, GO TO $L7$,
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0	$L9$	IF $R3<R1$, GO TO $L5$,
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0	$L10$	IF $R1<R3$, GO TO $L1$,
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0	$L11$	IF $R2<R1$, GO TO $L10$,
0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0	$L12$	$R1 \leftarrow R1-1$, $R2 \leftarrow R2-1$, $R3 \leftarrow R3-1$,
0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	$L13$	IF $0<R1$, GO TO $L12$,
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	$L14$	STOP.

Now we will show how to write down conditions on $x, s, Q, I, R_1, \dots, R_r, L_0, \dots, L_l$ such that the conditions have a solution if and only if M accepts x. Here r and l are constants, x is a parameter and s, Q, I, $R_1, \dots, R_r$, $L_0, \dots, L_l$ are unknowns. The methods of §2 then permit further translation of these conditions into exponential diophantine equations in the same unknowns and more unknowns.

The first four conditions will be (24), (25), (26) and (27). To force an arbitrary natural number R_j to have the form (22), we can make use of our binary relation $\preccurlyeq$, because the base Q itself is a power of 2. Condition (22) (and also $r_{j,t} < Q/2$ for all t, important in (35)) is implied by

$$R_j \preccurlyeq (Q/2 - 1)I \qquad (j = 1, \dots, r). \tag{29}$$

Next we want to force exactly one digit to be 1 in each column of the L matrix. Since

$l < Q$, by (25), we may use for this purpose the conditions

$$I = \sum_{i=0}^{l} L_i, \tag{30}$$

and

$$L_i \preccurlyeq I \qquad (i = 0, \dots, l). \tag{31}$$

In order to properly start the machine at location $L0$ we stipulate as a *starting condition*

$$1 \preccurlyeq L_0. \tag{32}$$

By means of GO TO commands we may suppose that the STOP command appears only once, say at the end of the program. Hence the condition for *stopping the machine*, after s steps, can be simply

$$L_l = Q^s. \tag{33}$$

For each command of type (15), Li GO TO Lk, we include a condition of type

$$QL_i \preccurlyeq L_k. \tag{34}$$

This forces the machine to transfer from location Li to location Lk, whenever line Li is executed.

To obtain the effect of a conditional transfer command we use a pair of conditions. For example to simulate the conditional transfer command (20), Li IF $Rj = 0$, GO TO Lk, we use

$$QL_i \preccurlyeq L_k + L_{i+1} \quad \text{and} \quad QL_i \preccurlyeq L_{i+1} + QI - 2R_j. \tag{35}$$

This works for the general case $k \neq i + 1$. If $k = i + 1$ use (34). A similar remark applies to the following conditional transfer commands. If $k \neq i + 1$, the command (16), Li IF $Rj < Rm$, GO TO Lk, can be simulated by

$$QL_i \preccurlyeq L_k + L_{i+1} \quad \text{and} \quad QL_i \preccurlyeq L_k + QI + 2R_j - 2R_m. \tag{36}$$

The command (19), Li IF $Rj \leq Rm$, GO TO Lk, can be simulated by

$$QL_i \preccurlyeq L_k + L_{i+1} \quad \text{and} \quad QL_i \preccurlyeq L_{i+1} + QI + 2R_m - 2R_j. \tag{37}$$

For each command of type (17) or (18) (or (21), if it is used), we must also include a condition directing the machine to take the next instruction, to execute $Li + 1$ after Li, i.e. $QL_i \preccurlyeq L_{i+1}$. Finally it is necessary to include also for each register Rj an equation in R_j which ensure that, at each time t, the tth Q-ary digit of R_j represents exactly the contents of register Rj. We call these *register equations*. They are the following:

$$R_j = QR_j + \sum_k QL_k - \sum_i QL_i + x \qquad (\text{for } j = 1), \tag{38}$$

and

$$R_j = QR_j + \sum_k QL_k - \sum_i QL_i \qquad (\text{for } j = 2, 3, \dots, r). \tag{39}$$

Here the k-sum is over all k for which the program has an instruction of the type $Lk \dots Rj \leftarrow Rj + 1 \dots$. The i-sum is taken over all i for which the program has an instruction of the form $Li \dots Rj \leftarrow Rj - 1 \dots$ (or one of type (21)). The register equation for $R1$ is different from that of the other registers because $R1$ contains x at the outset. In such respects the register equations can be easily modified to treat different input-output conventions. For example, in the case of computability of a function, $y = f(x)$, with input x in R_1 and output y in R_1, the register equation for R_1 would become

$$R_1 + yQ^{s+1} = QR_1 + \sum_k QL_k - \sum_i QL_i + x. \tag{40}$$

This completes the proof of the Davis, Putnam and Robinson theorem.

§4. Deterministic polynomial time equivalence. While it was of course not important in the above proof, register machines, limited to the above given types of commands, cannot add or multiply in so-called *polynomial time*, $s \le P(|x|)$, where P is a polynomial and $|x| = \ulcorner \log_2(x+1) \urcorner$ is the *length* of x.

To obtain register machines polynomial time equivalent to Turing machines it is sufficient to include two new commands:

$$Ln \quad Ri \leftarrow Ri + Rj, \tag{41}$$

$$Lp \quad Rj \leftarrow \ulcorner Rj/2 \urcorner. \tag{42}$$

From (41) and (42) one obtains the use of other (polynomial time) commands, e.g. $Ri \leftarrow 2Ri$, $Ri \leftarrow 0$, $Ri \leftarrow Rj$, $Ri \leftarrow 2Rj$, $Ri \leftarrow Rk + Rj$ and $Ri \leftarrow Rk + 2Rj$. For example, $Ri \leftarrow 0$ is obtained by iterating (42). We think of these commands as *macros*. Using the following trick

$$Ri \equiv 0 \pmod 2 \Leftrightarrow 2\ulcorner Ri/2 \urcorner \le Ri, \tag{43}$$

one obtains from (19) and (42) the use of the macro

$$\text{IF } Ri \equiv 0 \pmod 2, \text{ GO TO } Lk. \tag{44}$$

Then from (44) one obtains the macros *quotient*, $Ri \leftarrow \llcorner Ri/2 \lrcorner$, and *parity*, $Ri \leftarrow \mathrm{REM}(Ri, 2)$. These are sufficient for Minsky's [1967, pp. 170–172, 204–206] simulation of Turing machines by register machines (in polynomial time). For example we can now multiply in polynomial time, $Ri \leftarrow Rj \cdot Rk$ $(i \ne j \ne k \ne i)$:

$$\begin{array}{ll} & Ri \leftarrow 0, \\ L1 & \text{IF } Rk \equiv 0 \pmod 2, \text{ GO TO } L3, \\ & Ri \leftarrow Ri + Rj, \\ L3 & Rk \leftarrow \llcorner Rk/2 \lrcorner, \\ & Rj \leftarrow Rj + Rj, \\ & \text{IF } 0 < Rk, \text{ GO TO } L1. \end{array} \tag{45}$$

To simulate the new commands (41) and (42) with exponential diophantine equations, first replace (24) by

$$2^s(x+1) < Q/2. \tag{46}$$

For all the commands of type (41), include a sum $\sum_n QM_{j,n}$ in the register equation for Ri, where $M_{j,n}$ is a new unknown satisfying

$$(47) \qquad M_{j,n} \preccurlyeq R_j, \qquad M_{j,n} \preccurlyeq (Q-1)L_n, \qquad R_j \preccurlyeq (Q-1)(I-L_n) + M_{j,n}.$$

These three conditions (47) imply that

$$(48) \qquad M_{j,n} = \sum_{t=0}^{s} r_{j,t} l_{n,t} Q^t.$$

For all commands of type (42), $Lp\ Rj \leftarrow Rj - \lfloor Rj/2 \rfloor$, include a sum $-\sum_p QJ_{j,p}$ in the register equation for Rj, where each $J_{j,p}$ is a new unknown satisfying

$$(49) \quad 2J_{j,p} \preccurlyeq R_j, \qquad J_{j,p} \preccurlyeq (Q/2-1)L_p, \qquad R_j \preccurlyeq (Q-1)(I-L_p) + 2J_{j,p} + I.$$

The three conditions (49) imply that

$$(50) \qquad J_{j,p} = \sum_{t=0}^{s} \lfloor r_{j,t}/2 \rfloor\, l_{p,t} Q^t.$$

§5. Nondeterministic polynomial time equivalence. The new register machine proof can be applied also to the situation of nondeterministic polynomial time computation. This yields bounded diophantine characterizations of the class NP (first obtained by Adleman and Manders [1976]).

From the foregoing and Karp [1972] it is clear that we obtain nondeterministic register machines polynomial time equivalent to nondeterministic Turing machines by adding the following nondeterministic command:

$$(51) \qquad Ln \quad \text{BRANCH}\ (Li, Lj)$$

which causes transfer to location Li or Lj.

To simulate, with exponential diophantine equations, the nondeterministic command (51) we need only write

$$(52) \qquad QL_n \preccurlyeq L_i + L_j.$$

Putting $s \leqq P(|x|)$, estimating the sizes of the unknowns in terms of $P(|x|)$, combining the several $\preccurlyeq$ conditions into one $\preccurlyeq$ condition using (13), replacing (24) by (46) and using Theorem 7 (about exponentiation) from Adleman and Manders [1975], one obtains the following characterization of NP. A set A of nonnegative integers belongs to NP if and only if A can be represented in the form, $x \in A$ *iff*

$$(53) \qquad (\exists |x_0|, \ldots, |x_n| \le P(|x|))[x_0 \preccurlyeq x_1 \ \&\ Q(x, x_0, \ldots, x_n) = 0].$$

Here P and Q are polynomials with integer coefficients. The notation $\exists |y| \le P(|z|) \ldots$ is shorthand for $(\exists y)[|y| \le P(|z|) \& \ldots]$.

From (53), (10) and (5) one obtains also the following characterization of the class NP. A set $A \in NP$ if and only if A can be represented in the form, $x \in A$ *iff*

$$(54) \qquad (\exists |x_0|, \ldots, |x_n| \le P(|x|)[F(x, x_0, \ldots, x_n) = G(x, x_0, \ldots, x_n)],$$

where P is a polynomial and F and G are functions built up from $x, x_0, \ldots, x_n$ by the operations of addition, multiplication, and the logical "and" operation, &.

REFERENCES

L. Adleman and K. Manders [1975], *Computational complexity of decision procedures for polynomials*, ***16th Annual Symposium on Foundations of Computer Science (Berkeley, California, 1975)***, IEEE Computer Society, Long Beach, California, 1975, pp. 169–177. MR **57** #264.

——— [1976], *Diophantine complexity*, ***17th Annual Symposium on Foundations of Computer Science (Houston, Texas, 1976)***, IEEE Computer Society, Long Beach, California, 1976, pp. 81–88. MR **56** #7314.

Ja. M. Barzdin' [1963], *On a certain class of Turing machines (Minsky machines)*, ***Algebra i Logika***, vol. 1 (1962/63), no. 6, pp. 42–51. (Russian) MR **27** #2415.

E. Börger [1975], *Recursively unsolvable algorithmic problems and related questions re-examined*, ***⊨ ISILC Logic Conference, Proceedings of the International Summer Institute and Logic Colloquium, Kiel, 1974*** (G. H. Müller, A. Oberschelp and K. Potthoff, editors), Lecture Notes in Mathematics, vol. 499, Springer-Verlag, Berlin, 1975, pp. 10–24. MR **58** #10355.

M. Davis [1953], *Arithmetical problems and recursively enumerable predicates*, this Journal, vol. 18 (1953), pp. 33–41. MR **14**, 1052.

——— [1958], ***Computability and unsolvability***, McGraw-Hill, New York, 1958. MR **23** #A1525.

——— [1973], *Hilbert's tenth problem is unsolvable*, ***American Mathematical Monthly***, vol. 80 (1973), pp. 233–269. MR **47** #6465.

——— [1974], ***Computability***, Lecture Notes, Courant Institute of Mathematical Sciences, New York University, New York, 1974. MR **50** #77.

M. Davis, Y. V. Matijasevič and J. Robinson [1976], *Hilbert's tenth problem. Diophantine equations: positive aspects of a negative solution*, ***Mathematical developments arising from Hilbert problems***, Proceedings of Symposia in Pure Mathematics, vol. 28, American Mathematical Society, Providence, Rhode Island, 1976, pp. 323–378. MR **55** #5522.

M. Davis, H. Putnam and J. Robinson [1961], *The decision problem for exponential diophantine equations*, ***Annals of Mathematics***, ser. 2, vol. 74 (1961), pp. 425–436. MR **24** #A3061.

N. J. Fine [1947], *Binomial coefficients modulo a prime*, ***American Mathematical Monthly***, vol. 54 (1947), pp. 589–592. MR **9**, 331.

M. Hausner [1983], *Applications of a simple counting technique*, ***American Mathematical Monthly***, vol. 90 (1983), pp. 127–129.

K. Hirose and S. Iida [1973], *A proof of negative answer to Hilbert's tenth problem*, ***Proceedings of the Japan Academy***, vol. 49 (1973), pp. 10–12. MR **58** #27750.

J. P. Jones [1974], *Recursive undecidability—an exposition*, ***American Mathematical Monthly***, vol. 81 (1974), pp. 724–738. MR **50** #9568.

——— [1978], *Three universal representations of recursively enumerable sets*, this Journal, vol. 43 (1978), pp. 335–351. MR **58** #16226.

——— [1982]. *Universal diophantine equation*, this Journal, vol. 47 (1982), pp. 549–571.

J. P. Jones and Y. V. Matijasevič [1982], *Exponential diophantine representation of recursively enumerable sets*, ***Proceedings of the Herbrand Symposium, Logic Colloquium '81*** (J. Stern, editor), Studies in Logic and the Foundations of Mathematics, vol. 107, North-Holland, Amsterdam, 1982, pp. 159–177.

R. M. Karp [1972], *Reducibility among combinatorial problems*, ***Complexity of computer computations*** (R. E. Miller and J. W. Thatcher, editors), Plenum Press, New York, 1972, pp. 85–103. MR **51** #14644.

J. Lambek [1961], *How to program an infinite abacus*, ***Canadian Mathematical Bulletin***, vol. 4 (1961), pp. 295–302. MR **24** #A2532.

E. Lucas [1878a], *Sur les congruences des nombres eulériens et des coefficients différentiels des fonctions trigonométriques suivant un module premier*, ***Bulletin de la Société Mathématique de France***, vol. 6 (1877/78), pp. 49–54.

——— [1878b], *Théorie des fonctions numériques simplement périodiques*, ***American Journal of Mathematics***, vol. 1 (1878), pp. 184–240; English translation available from the Fibonacci Association, San Jose University, San Jose, California, 1969.

Y. V. Matijasevič [1970], *Enumerable sets are diophantine*, ***Doklady Akademii Nauk SSSR***, vol. 191 (1970), pp. 279–282; English translation with addendum, ***Soviet Mathematics: Doklady***, vol. 11 (1970), pp. 354–357. MR **41** #3390.

——— [1971a], *Diophantine representation of enumerable predicates*, ***Izvestija Akademii Nauk SSSR Serija Matematičeskaja***, vol. 35 (1971), pp. 3–30; English translation, ***Mathematics of the USSR—Izvestija***, vol. 5 (1971), pp. 1–28. MR **43** #54.

——— [1971b], *Diophantine representation of the set of prime numbers*, ***Doklady Akademii Nauk SSSR***, vol. 196 (1971), pp. 770–773; English translation, ***Soviet Mathematics: Doklady***, vol. 12 (1971), pp. 249–254. MR **43** #1921.

——— [1971c], *On recursive unsolvability of Hilbert's tenth problem*, ***Proceedings of the Fourth International Congress for Logic, Methodology and Philosophy of Science (Bucharest, 1971)***, Studies in Logic and the Foundations of Mathematics, vol. 74, North-Holland, Amsterdam, 1973, pp. 89–110. MR **57** #5711.

——— [1972], *Diophantine sets*, ***Uspehi Matematičeskih Nauk***, vol. 27 (1972), no. 5 (167), pp. 185–222; English translation, ***Russian Mathematical Surveys***, vol. 27 (1972), no. 5, pp. 124–164. MR **56** #109.

——— [1974], *The existence of noneffectizable estimates in the theory of exponential diophantine equations*, ***Zapiski Naučnyh Seminarov Leningradskogo Otdelenija Matematičeskogo Instituta im. V.A. Steklova (LOMI) Akademii Nauk SSSR***, vol. 40 (1974), pp. 77–93; English translation, ***Journal of Soviet Mathematics***, vol. 8 (1977), pp. 299–311. MR **51** #10225.

——— [1976], *A new proof of the theorem on exponential diophantine representation of enumerable sets*, ***Zapiski Naučnyh Seminarov Leningradskogo Otdelenija Matematičeskogo Instituta im. V.A. Steklova (LOMI) Akademii Nauk SSSR***, vol. 60 (1976), pp. 75–89; English translation, ***Journal of Soviet Mathematics***, vol. 14 (1980), pp. 1475–1486. MR **58** #27402.

——— [1977], *A class of primality criteria formulated in terms of the divisibility of binomial coefficients*, ***Zapiski Naučnyh Seminarov Leningradskogo Otdelenija Matematičeskogo Instituta im. V.A. Steklova (LOMI) Akademii Nauk SSSR***, vol. 67 (1977), pp. 167–183; English translation, ***Journal of Soviet Mathematics***, vol. 16 (1981), pp. 874–885. MR **57** #3060.

——— [1979], *Algorithmic unsolvability of exponential diophantine equations in three unknowns*, ***Studies in the Theory of Algorithms and Mathematical Logic*** (A. A. Markov and V. I. Homič, editors), "Nauka", Moscow, 1979, pp. 69–78; English translation, to appear in ***Selecta Mathematica Sovietica***, vol. 3 (1983), no. 3. MR **81f**: 03055.

Y. Matijasevič and J. Robinson [1975], *Reduction of an arbitrary diophantine equation to one in 13 unknowns*, ***Acta Arithmetica***, vol. 27 (1975), pp. 521–553. MR **52** #8033.

Z. A. Melzak [1961], *An informal arithmetical approach to computability and computation*, ***Canadian Mathematical Bulletin***, vol. 4 (1961), pp. 279–294. MR **27** #1364.

M. Minsky [1961], *Recursive unsolvability of Post's problem of "tag" and other topics in the theory of Turing machines*, ***Annals of Mathematics***, ser. 2, vol. 74 (1961), pp. 437–455. MR **25** #3825.

——— [1967], ***Computation: Finite and infinite machines***, Prentice-Hall, Englewood Cliffs, New Jersey, 1967. MR **50** #9050.

J. Robinson [1952], *Existential definability in arithmetic*, ***Transactions of the American Mathematical Society***, vol. 72 (1952), pp. 437–449. MR **14**, 4.

——— [1969], *Diophantine decision problems*, ***Studies in number theory*** (W. J. LeVeque, editor), MAA Studies in Mathematics, vol. 6, Mathematical Association of America, Buffalo, New York (distributed by Prentice-Hall, Englewood Cloiffs, New Jersey), 1969, pp. 76–116. MR **39** #5364.

J. C. Shepherdson and H. E. Sturgis [1963], *Computability of recursive functions*, ***Journal of the Association for Computing Machinery***, vol. 10 (1963), pp. 217–255. MR **27** #1359.

C. Smoryński [1972], ***Notes on Hilbert's tenth problem: an introduction to unsolvability***. Vol. 1, Department of Mathematics, University of Illinois at Chicago Circle, Chicago, Illinois, 1972.

UNIVERSITY OF CALGARY
CALGARY, ALBERTA, CANADA T2N 1N4

STEKLOV INSTITUTE OF MATHEMATICS
ACADEMY OF SCIENCES OF THE USSR
LENINGRAD, USSR 191011