Proof of Recursive Unsolvability of Hilbert's Tenth Problem

J. P. JONES, *University of Calgary, Calgary, Alberta, Canada, T2N 1N4*

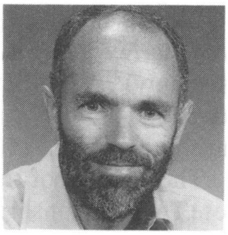
Y. V. MATIJASEVIČ, *Steklov Math. Institute, Leningrad, USSR, 191011*

JAMES P. JONES received his Ph.D. in 1968 at the University of Washington, Seattle. His supervisor was R. W. Ritchie. He is presently in the Department of Mathematics of the University of Calgary, Alberta, Canada.

He has spent leaves in the Steklov Mathematical Institute of the Academy of Sciences of the U.S.S.R., Leningrad, the Tata Institute of Fundamental Research, the University of California, Berkeley and Academia Sinica, Beijing.

He has given lectures in Moscow State University, the Institute of Cybernetics in Tallinn, U.S.S.R., the Steklov Mathematical Institute in Leningrad, U.S.S.R., and in the Main Computing Center of the Academy of Sciences of the U.S.S.R. in Riga. He has also lectured at the Stefan Banach Center in Warsaw and at universities in England, France, Italy, Germany, China and Nepal.

His current research area is logic and number theory.

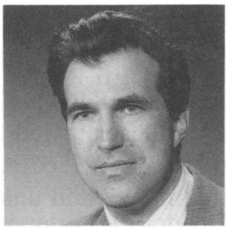


YURI V. MATIJASEVIČ received his Candidate of Physical and Mathematical Sciences (Ph.D.) degree in the Steklov Mathematical Institute in Leningrad, in 1970. His supervisors were N. A. Shanin and G. S. Tseitin. The degree was given for constructing a semigroup with three relations and an unsolvable word problem.

Matijosevič received his Doctor Sci. degree also in the Steklov Mathematical Institute in Leningrad, in 1973. This was for the solution of Hilbert's Tenth Problem in 1970. Then his supervisor was S. U. Maslov.

Matijasevič has given lectures in Canada, France, Hungary, Romania and the U.S. In the U.S. he has lectured in the Courant Institute, M.I.T., Boston University, Berkeley and Stanford. In Canada he has lectured in Vancouver, Calgary, London Ontario, Banff, Kingston, Quebec, Thunder Bay and Montreal. In the U.S.S.R. he has lectured in Leningrad, Moscow, Kiev, Tallinn, Kishinev and Tbilisi.

His current research area is logic and number theory.



The purpose of the present paper is to give a modern complete proof of unsolvability of the Tenth Problem of Hilbert. Our intention is to give the shortest proof known today, one which takes into account all the simplifications found since the problem was solved by the second author in [1970].

The Tenth Problem of Hilbert is the problem of solvability of diophantine equations. As originally formulated by Hilbert [1900], it was to find an algorithm to decide whether a polynomial equation in several variables, $P(x_1, x_2, \ldots, x_n) = 0$, has a solution in integers.

The problem can also be formulated in terms of existence of solutions in natural numbers (nonnegative integers). These two forms of Hilbert's Tenth problem are equivalent. An equation $P(x_1, x_2, \ldots, x_n) = 0$ has a solution in integers if and only if the product $\Pi P(\pm x_1, \pm x_2, \ldots, \pm x_n) = 0$ has a solution in natural numbers. Also, from Lagrange's Four Squares Theorem [1770], $P(x_1, x_2, \ldots, x_n) = 0$

has a solution in natural numbers iff $P(x_1^2 + y_1^2 + u_1^2 + v_1^2,\ x_2^2 + y_2^2 + u_2^2 + v_2^2, \dots, x_n^2 + y_n^2 + u_n^2 + v_n^2) = 0$ has a solution in integers.

What will be proved here is the algorithmic unsolvability of Hilbert's Tenth Problem, i.e., the nonexistence of an algorithm, over natural numbers. This will be done in the normal way, by reducing another unsolvable problem to Hilbert's Tenth. Actually we will reduce all unsolvable problems to Hilbert's Tenth. To explain we need some definitions.

Let A be a set or relation on the natural numbers. The set A is called *recursively enumerable* (*r.e.*) if A is the exact range of a recursive function (or if A is empty). A set is called *recursive* if its characteristic function is a recursive function. A function f is said to be *recursive* if it is computable by a Turing machine or register machine (see §4).

The concept of recursively enumerable set is more general than that of recursive set. The relationship between the two concepts is also important here. A set A is recursive if and only if A and its complement A' are both r.e. sets. Thus every recursive set is recursively enumerable. However there exist r.e. sets which are non-recursive. Hence the negative solution to Hilbert's Tenth problem will follow immediately from the following theorem:

THEOREM 1. *Every r.e. relation* $A(a_1, a_2, \dots, a_m)$ *can be represented in the form*

$$A(a_1, \dots, a_m) \Leftrightarrow (\exists x_1, \dots, x_n)[P(a_1, \dots, a_m, x_1, \dots, x_n) = 0]. \qquad (1)$$

Here $P(a_1, \dots, a_m, x_1, \dots, x_n)$ is a polynomial with integer coefficients which depend on A. The Theorem is that for each r.e. set A, a polynomial P exists such that (1) is satisfied for all values of $a_1, a_2, \dots, a_m$ (the *parameters*). The variables $x_1, x_2, \dots, x_n$, correspond to the *unknowns* in our equation. All the variables, parameters and unknowns, range over the same set, nonnegative integers.

A set or relation $A(a_1, \dots, a_m)$ in the nonnegative integers, definable in the form (1), is called a *diophantine* set. Thus Theorem 1 asserts that every r.e. set is diophantine. The converse also holds trivially: every diophantine set is r.e. Thus Theorem 1 asserts the equality of two collections of sets.

Theorem 1 implies the nonexistence of an algorithm for Hilbert's Tenth Problem because it reduces the decision problem for every r.e. set, to some instance of Hilbert's Tenth. If Hilbert's Tenth were solvable, then every r.e. set would be recursive. However, as we mentioned, there exist r.e. sets which are nonrecursive. Taking the set A in Theorem 1 to be such a set, one sees that there can exist no algorithm for Hilbert's Tenth Problem.

This argument seems completely satisfying to us: however, for those who wish it we give an additional argument after the proof of Theorem 1 (section 5).

Theorem 1 also implies the algorithmic unsolvability of Hilbert's Tenth Problem. This is a slightly stronger conclusion than recursive unsolvability. When we come to this conclusion on the basis of Theorem 1, we are implicitly using the so-called Church–Turing Thesis. This is the statement that every computable function is Turing computable. It is called a thesis rather than a theorem because

Work supported by Natural Sciences and Engineering Research Council of Canada research grant A4525, the NSERC Program of International Scientific Exchange Awards, the Queen's–Steklov Exchange Program between Canada and the USSR and the U.S. National Academy of Sciences.

it is apparently unprovable. It is somewhat in the nature of a definition, or proposal, that we identify the intuitive concept of computability with the precise mathematical formalization, Turing computability, register machines or General Recursiveness.

Today the Church–Turing Thesis is widely accepted. If an algorithm exists to solve Hilbert's Tenth problem, then Theorem 1 implies that it would have to lie beyond the present concept of algorithm, beyond Turing machines, beyond register machines, beyond Markov algorithms and indeed beyond all known formalizations of algorithm (all of which have been proven equivalent).

Theorem 1 implies for example that an algorithm solving Hilbert's Tenth Problem would also solve the word problem for groups, the halting problem for Turing machines and all other known r.e. unsolvable problems. Needless to say, no example of such an algorithm is known.

If the reader wishes to know more about the theory of computability, he or she is directed to the books of Davis [1958], Minsky [1967], Rogers [1967], the second author's paper [1984] or the first author's [1974] MONTHLY paper.

The unsolvability of Hilbert's Tenth problem was originally proved in two steps. Step 1 was taken by Martin Davis, H. Putnam and Julia Robinson [1961] who obtained an exponential form of Theorem 1. Here P was not a polynomial but contained an exponential function, $y = a^x$. In this form Theorem 1 can be interpreted as saying that every r.e. set is exponential diophantine.

The second step in the solution of Hilbert's Tenth Problem was to show that the exponential relation itself $y = a^x$ is diophantine. This difficult last step was taken by the second author [1970]. This proved Theorem 1 and solved Hilbert's Tenth Problem in the negative.

To show that the exponential function $y = a^x$ is diophantine, divisibility properties of the sequence of Fibonacci numbers were used in [1970]. Subsequently it was seen how to do this using the sequence of solutions of the Pell equation. Proofs of Chudnovsky [1970], Davis [1971] [1973] and Kosovskii [1971] all use the sequence of solutions of Pell's equation.

Today the Pell equation gives the simplest known proof. We will also use the Pell equation here. Concerning this part of the proof, considerable credit goes to Martin Davis and Julia Robinson. Many simplifications in the present proof can be traced to their discoveries.

By way of small new improvements in the Pell equation part of the present proof, we mention (for experts), that in the main lemma on diophantine representation of the sequence of solutions of Pell's equation (Lemma 2.27), use of the Chinese remainder theorem has been eliminated. (Davis [1973] p. 246.) Those who know the subject will also find that in this Pell equation part of the proof we eliminated also the use of the axiom of existence of infinitely many solutions to the general Pell equation,

$$(\forall d)(\exists x, y)\left[d \neq \square \rightarrow x^2 - dy^2 = 1 \wedge 0 < y\right]. \tag{2}$$

This assumption (or axiom) was used in Matijasevič–Robinson [1975] (pp. 532–533). Of course the axiom is true. From a certain point of view its use doesn't matter. However, if one is concerned with formalizability of the proof in axiomatic theories, then use of this axiom is a deficiency. So we are pleased to have eliminated it.

When d is an arbitrary non-square, as is the case in (2) ($d \neq \square$), then it is difficult to prove statement (2) from a restricted set of first order axioms. For example statement (2) cannot be proved in the theory known as Bounded Arithmetic; Peano's axioms with the induction axiom

$$A(0) \wedge (\forall x)[A(x) \rightarrow A(x+1)] \rightarrow (\forall x)[A(x)], \tag{3}$$

restricted to instances where all quantified variables in the $A(x)$ are bounded by polynomials in x.

Indeed, whether the whole of the proof of unsolvability of Hilbert's Tenth problem can be formalized in Bounded Arithmetic is an open problem. J. Paris and C. Dimitracopoulos [1982] showed that this would be the case if we add one more axiom (stronger than (2)). But without this axiom the answer is not known.

This formalization problem is important because if the proof of Hilbert's Tenth Problem can be formalized in Bounded Arithmetic, then, from an idea of A. J. Wilkie, $NP = co - NP$ follows. This is why we mention that in this paper axiom (2) is used only for d of the special form, $d = a^2 - 1$. For these d axiom (2) is provable in Bounded Arithmetic. This increases the likelihood that the present proof can be formalized there.

Besides its use in formalization, the proof here can also be used as the main link in a proof that every Turing computable function is recursive. We give a method for encoding computations of arbitrary register machines which greatly simplifies the tedious arithmetization usually involved with this procedure. This method of coding we call *bit masking*. It uses a famous theorem of Lucas [1878] on congruences in binomial coefficients this method was worked out in our [1984]. Originally we made use of a classical theorem of Kummer [1852], on the power of a prime dividing a binomial coefficient. Later we discovered the simpler proof using Lucas' Theorem (see Lemma 3.10).

Nearly the whole of the present proof of Hilbert's Tenth Problem is now number theory. In fact it is classical number theory. Section 2, on the sequence of solutions of Pell's equation, belongs to the Lucas–Lehmer theory of recurrent sequences. In the terminology of Lehmer [1930], this is the study of the Lucas sequences U_n and V_n where $P = 2a$, $Q = 1$, $D = 4(a^2 - 1)$ and $R = 4a^2$. With these values of the parameters, U_n and V_n correspond to the sequence of solutions of Pell's equation, $x^2 - dy^2 = 1$ with $d = a^2 - 1$. Here $U_n = Y_a(n)$ and $V_n = 2 \cdot X_a(n)$ where $X_a(n)$ and $Y_a(n)$ are the sequence of solutions of $x^2 - (a^2 - 1)y^2 = 1$, (see §2).

This theory is very old. Though the multiplication by 2 makes a small difference, it is not difficult to interpret $Y_a(n)$ and $X_a(n)$ in terms of U_n and V_n and so to see that many of the theorems go back to Lucas [1878] and Lehmer [1930]. Some modern theorems are due to Julia Robinson [1952] [1969]. In this connection it is interesting that while Lucas preferred the functions U_n and V_n, Julia Robinson preferred the sequences $Y_a(n)$ and $X_a(n)$.

§1 Diophantine Sets. A diophantine equation is a polynomial equation, $P(a_1, \ldots, a_m, x_1, \ldots, x_n) = 0$, in several variables, with integer coefficients. The variables are divided into *parameters* $a_1, a_2, \ldots, a_m$, and *unknowns*, $x_1, x_2, \ldots, x_n$. All the variables, parameters and unknowns, range over the set of nonnegative integers, $0, 1, 2, \ldots$.

In the classical theory of diophantine equations one begins with an equation and asks for values of the parameters for which there exists a solution. In this proof we turn the usual procedure around. We start with the solution and search for the equation. We begin with a relation $A(a_1, a_2, \ldots, a_m)$ and we look for a polynomial $P(a_1, \ldots, a_m, x_1, \ldots, x_n)$ defining it in the sense of (1).

DEFINITION 1.1. A relation $A(a_1, a_2, \ldots, a_m)$ is *diophantine* if there exists a polynomial $P(a_1, \ldots, a_m, x_1, \ldots, x_n)$ such that for all values of $a_1, \ldots, a_m$, (the parameters)

$$A(a_1, \ldots, a_m) \Leftrightarrow (\exists x_1, x_2, \ldots, x_n)[P(a_1, \ldots, a_m, x_1, \ldots, x_n) = 0]. \quad (1.1)$$

This definition which is for relations, will also do for functions. A *function* will be said to be diophantine if its graph is diophantine. Below are examples of diophantine functions and relations. Most will be used in the proof. These examples include the elementary relations of *order* $\leqslant$, *divisibility* $a|b$, and *congruence*, $a \equiv b \pmod c$.

$$a \leqslant b \Leftrightarrow (\exists x)[a + x = b], \quad (1.2)$$

$$a|b \Leftrightarrow (\exists x)[ax = b], \quad (1.3)$$

$$a \equiv b \pmod c \Leftrightarrow (\exists x)[a = b + cx \text{ or } a = b - cx]. \quad (1.4)$$

Disjunctions and conjunctions as occur in connection with (1.4) can be dealt with using

$$A = 0 \text{ or } B = 0 \Leftrightarrow A \cdot B = 0, \qquad A = 0 \,\&\, B = 0 \Leftrightarrow A^2 + B^2 = 0. \quad (1.5)$$

In the case of conjunctions, it is necessary to rename variables which occur in both A and B, for example $\exists x A(x) = 0 \wedge \exists x B(x) = 0$ is equivalent to $\exists x \exists y[A(x) = 0 \wedge B(y) = 0]$.

From (1.5) it follows that conjunctions and disjunctions of Diophantine relations (but not negations) are diophantine. Proceeding in this way, we can prove that a great many relations are diophantine. A good is the relation of *coprimality* (a is *relatively prime* to b, $(a, b) = 1$) written in this paper as $a \perp b$. This relation can be seen to be diophantine using (1.5) together with

$$a \perp b \Leftrightarrow (\exists x, y)[ax - by = 1 \text{ or } ax - by = -1]. \quad (1.6)$$

Other examples of diophantine functions are the *remainder function*, $r = rem(a, b)$ (r is the remainder after a is divided by b), and the *quotient function*, $q = quo(a, b)$, meaning q is the quotient when a is divided by b. (This is the same as the integer part function, $q = \lfloor a/b \rfloor$.) These functions can be seen to be diophantine from

$$r = rem(a, b) \leftrightarrow r \equiv a \pmod b \text{ and } r < b, \quad (1.7)$$

$$q = quo(a, b) \leftrightarrow 0 \leqslant a - qb < b. \quad (1.8)$$

Proceeding in this way, defining new relations from old, using, often (1.5) we will show that a great many relations are diophantine. The most important tool in this process will be the sequence of solutions of the Pell equation.

§2. The Pell equation. A general Pell equation is an equation of the form

$$x^2 - dy^2 = 1 \qquad (d \neq \square). \quad (2.1)$$

where d is a constant and x and y are unknowns. (A hyperbola in the x, y plane.)

When $d = \square$ (d is a square), then the Pell equation (2.1) has only the trivial solution, $(1, 0)$. So one always assumes that $d \neq \square$ (d is not a square).

Because $x^2 - dy^2$ factors over the reals, $x^2 - dy^2 = (x + \sqrt{d}\,y)(x - \sqrt{d}\,y)$, it is natural to work in the integral domain $\mathbb{Z}[\sqrt{d}]$ consisting of real numbers of the form $\alpha = x + y\sqrt{d}$, $x, y \in \mathbb{Z}$. Since $\sqrt{d}$ is irrational, the integers x and y are unique. They are called the *components* of α.

When $\alpha = x + y\sqrt{d}$, the number $\bar{\alpha} = x - y\sqrt{d}$ is known as the *conjugate* of α. The number $N(\alpha) = \alpha\bar{\alpha} = x^2 - dy^2$ is called the *norm* of α. Since $N(\alpha) = x^2 - dy^2$, the integer solutions to the Pell equation are just the components x, y of reals α such that $N(\alpha) = 1$.

$N(\alpha) = 1$ and $N(\alpha) = \alpha \cdot \bar{\alpha}$ imply $\alpha^{-1} = \bar{\alpha}$, the conjugate of α is equal to the inverse. Also $\alpha = x + \sqrt{d} \cdot y$ and $1 \leqslant \alpha$ implies $1 \leqslant x$ and $0 \leqslant y$. To see this observe that from $0 < \alpha^{-1} \leqslant 1$, $\alpha + \bar{\alpha} = 2x$ and $\alpha - \bar{\alpha} = 2y\sqrt{d}$ we have $\frac{1}{2} \leqslant x$ and $0 \leqslant y$. Then from $x^2 = 1 + dy^2$ we have $1 \leqslant x$. Hence the inequality $1 \leqslant \alpha$ implies that the components x and y of α are nonnegative.

From $\overline{\alpha\beta} = \bar{\alpha} \cdot \bar{\beta}$ we have $N(\alpha\beta) = \alpha\beta \cdot \overline{\alpha\beta} = \alpha \cdot \beta \cdot \bar{\alpha} \cdot \bar{\beta} = \alpha \cdot \bar{\alpha} \cdot \beta \cdot \bar{\beta} = N(\alpha) \cdot N(\beta)$. Thus if $N(\alpha) = 1$ and $N(\beta) = 1$, then $N(\alpha \cdot \beta) = 1$. So the product of two reals representing solutions of the Pell equation represents again a solution. We also have $N(\bar{\alpha}') = N(\alpha)$.

Now let $\alpha = x_1 + y_1\sqrt{d}$ and $\beta = x_2 + y_2\sqrt{d}$ be two solutions of the Pell equation, with x_1, y_1, x_2, y_2 integers, $N(\alpha) = 1$ and $N(\beta) = 1$. Suppose $1 \leqslant \alpha$ and $1 \leqslant \beta$ hold. As we have seen, x_1, y_1, x_2, y_2 must then be nonnegative. So from the equations $x_1^2 = dy_1^2 + 1$ and $x_2^2 = dy_2^2 + 1$, it follows that $x_1 < x_2$ iff $y_1 < y_2$. Thus $1 \leqslant \alpha < \beta$ holds iff both $1 \leqslant x_1 < x_2$ and $0 \leqslant y_1 < y_2$ hold. So the set of reals α for which $N(\alpha) = 1$ and $1 < \alpha$ hold, is a well ordered set. If this set is nonempty, if there exists a nontrivial solution to the Pell equation, then there must exist a least real α such that $1 < \alpha$ and $N(\alpha) = 1$. This real is called the *generator*. Its components are called the *fundamental solution*.

The powers of the generator α generate all solutions to (2.1). To see this, observe first that $N(\alpha) = 1$ implies $N(\alpha^n) = N(\alpha)^n = 1$, so that powers of α represent solutions. To see that these powers of α give all solutions of (2.1), observe that if $N(\beta) = 1$ and $1 \leqslant \beta$, then $\exists n$ such that $\alpha^n \leqslant \beta < \alpha^{n+1}$. From $1 \leqslant \beta \cdot \alpha^{-n} < \alpha$ and $N(\beta\alpha^{-n}) = N(\beta)N(\alpha^{-1})^n = N(\beta)N(\bar{\alpha})^n = N(\beta)N(\alpha)^n = 1$, it then follows that $\beta \cdot \alpha^{-n} = 1$. Hence $\beta = \alpha^n$. Therefore β can be obtained as a power of α.

It is not entirely trivial to show that for arbitrary $d \neq \square$ there always exists a nontrivial solution to the Pell equation (2.1). But when d is a nonsquare of the special form $d = a^2 - 1$, then it is easy to prove this and hence the existence of infinitely many solutions. The special Pell equation with $d = \alpha^2 - 1$,

$$x^2 - (a^2 - 1)y^2 = 1, \tag{2.2}$$

has the fundamental solution $(x, y) = (a, 1)$. So the generator is $a + \sqrt{a^2 - 1}$. By the preceding, the powers of this real then give all solutions,

$$X_a(n) + Y_a(n) \cdot \sqrt{a^2 - 1} = \left(a + \sqrt{a^2 - 1}\right)^n. \tag{2.3}$$

Let $x = X_a(n)$ and $y = Y_a(n)$ denote this sequence of solutions to (2.2), defined by (2.3). Then $X_a(n)$ and $Y_a(n)$ are strictly increasing functions of n. This is evident from (2.9) below.

Taking the conjugate of both sides of (2.3) one finds that the sequences $X_a(n)$ and $Y_a(n)$ are also definable from the conjugate of the generator, $a - \sqrt{a^2 - 1}$,

$$X_a(n) - Y_a(n)\sqrt{a^2 - 1} = \left(a - \sqrt{a^2 - 1}\right)^n. \tag{2.4}$$

As we have mentioned, $\bar{\alpha} = \alpha^{-1}$, the conjugate of the generator is equal to the inverse,

$$a - \sqrt{a^2 - 1} = \left(a + \sqrt{a^2 - 1}\right)^{-1}. \tag{2.5}$$

This implies that most of the identities which hold for the sequence of solutions of the Pell equation also hold for negative values of parameters. For example from the identities $(a + \sqrt{d})^{n+m} = (a + \sqrt{d})^n(a + \sqrt{d})^m$, $(a + \sqrt{d})^{n-m} = (a + \sqrt{d})^n(a + \sqrt{d})^{-m}$, using (2.3), (2.4) and (2.5) one obtains the identity

$$X_a(n \pm m) + Y_a(n \pm m)\sqrt{d} = \left(X_a(n) + Y_a(n)\sqrt{d}\right)\left(X_a(m) \pm Y_a(m)\sqrt{d}\right). \tag{2.6}$$

Taking the rational and irrational parts of (2.6), one obtains the Addition Equations (Lucas). Here the ± signs correspond.

$$X_a(n \pm m) = X_a(n)X_a(m) \pm dY_a(n)Y_a(m), \tag{2.7}$$

$$Y_a(n \pm m) = Y_a(n)X_a(m) \pm X_a(n)Y_a(m). \tag{2.8}$$

These hold only for $a \geqslant 2$ but if we define $X_1(n) = 1$ and $Y_1(n) = n$, then (2.8) will hold for $a = 1$. Putting $m = 1$ in (2.7) and (2.8), we see that also, as a special case of (2.7), (2.8)

$$X_a(n + 1) = aX_a(n) + dY_a(n), \qquad Y_a(n + 1) = aY_a(n) + X_a(n), \tag{2.9}$$

$$X_a(n - 1) = aX_a(n) - dY_a(n), \qquad Y_a(n - 1) = aY_a(n) - X_a(n). \tag{2.10}$$

Adding pairs of equations (2.9) and (2.10), one obtains representations of X and Y as *Lucas Sequences*, that is, sequences satisfying a second order linear recurrence.

$$X_a(0) = 1, \qquad X_a(1) = a, \qquad X_a(n + 1) = 2aX_a(n) - X_a(n - 1). \tag{2.11}$$

$$Y_a(0) = 0, \qquad Y_a(1) = 1, \qquad Y_a(n + 1) = 2aY_a(n) - Y_a(n - 1). \tag{2.12}$$

From the addition equations, (2.7) and (2.8), together with defining equation (2.2), one can derive the *Double Angle Formulas* (Lucas)

$$X_a(2n) = 2X_a(n)^2 - 1, \tag{2.13}$$

$$Y_a(2n) = 2X_a(n)Y_a(n). \tag{2.14}$$

Generally we are interested in the X_a and Y_a sequences only for $a \geqslant 2$. However as we mentioned, when $a = 1$ they can be defined in a natural way satisfying (2.8), (2.11) and (2.12). We can define $X_1(n) = 1$ and $Y_1(n) = n$.

For fixed n, the function $Y_a(n)$ is a polynomial in a. This can be seen from (2.12). The degree is $n - 1$. From this, that $Y_a(n)$ is a polynomial in a, one obtains

Congruence Rule, $$Y_a(n) \equiv Y_b(n) \pmod{a - b}. \tag{2.15}$$

The Congruence Rule holds for a, $b \geqslant 1$. Putting $b = 1$ in (2.15) and using $Y_1(n) = n$ we obtain the special congruence rule of Julia Robinson [1952]:

$$Y_a(n) \equiv n \pmod{a - 1}. \tag{2.16}$$

The Lucas equations (2.11), (2.12) also enable one to derive bounds on the size of the X_a and Y_a sequences. For example, it is easy to derive the following bound on $Y_a(n)$:

$$(2a-1)^n \leqslant Y_a(n+1) < (2a)^n. \tag{2.17}$$

This inequality, which holds for $1 \leqslant a$, $1 < n$, shows that the $Y_a(n)$ sequence grows exponentially in n. These sequences can also be more directly related to exponentiation. The following congruence was obtained by Julia Robinson [1952]:

$$X_a(n) - (a-k)Y_a(n) \equiv k^n \pmod{2ak - k^2 - 1}. \tag{2.18}$$

Proof. We will show that the congruence holds for all $n \geqslant 0$ and all $k \geqslant 0$. It is easy to check for $n = 0$ and $n = 1$. So proceeding by induction on n, using recurrence equations (2.11) and (2.12), we obtain

$$\begin{aligned} X_a(n+1) - (a-k)Y_a(n+1) &= 2a \cdot X_a(n) - X_a(n-1) \\ &\quad -(a-k)[2a \cdot Y_a(n) - Y_a(n-1)] \\ &= 2a[X_a(n) - (a-k)Y_a(n)] \\ &\quad -[X_a(n-1) - (a-k)Y_a(n-1)] \\ &\equiv 2ak^n - k^{n-1} = k^{n-1}(2ak-1) \\ &= k^{n-1}(2ak - k^2 - 1 + k^2) \equiv k^{n-1}(0 + k^2) \\ &= k^{n-1}k^2 = k^{n+1} \pmod{2ak - k^2 - 1}. \end{aligned}$$

Next we prove a divisibility property which we will need in the proof of (2.20).

$$n|m \leftrightarrow Y_a(n)|Y_a(m). \tag{2.19}$$

Proof. From the Addition Equation (2.8) we have (dropping the subscript a), $Y(k \pm n) = Y(k)X(n) \pm X(k)Y(n) \equiv Y(k)X(n) \pmod{Y(n)}$. But $Y(n) \perp X(n)$ ($Y(n)$ and $X(n)$ are relatively prime), by (2.2). So $Y(n)|Y(k \pm n)$ iff $Y(n)|Y(k)$. Now let $m = ni + r$ where $0 \leqslant r < n$. Then $0 \leqslant Y(r) < Y(n)$. Also $Y(n)|Y(m)$ iff $Y(n)|Y(ni + r)$ iff $Y(n)|Y(r)$. Hence $Y(n)|Y(m)$ iff $r = 0$, i.e. $Y(n)|Y(m)$ iff $n|m$.

First Step Down Lemma 2.20. $Y_a^2(n)|Y_a(m) \leftrightarrow n \cdot Y_a(n)|m$. $(1 \leqslant a)$.

Proof. By using identity (2.3), twice, it is easy to prove that for any j

$$X_a(nj) + Y_a(nj)\sqrt{d} = \left(X_a(n) + Y_a(n)\sqrt{d}\right)^j. \tag{2.21}$$

Now expand the right side of (2.21) and take the irrational part to get

$$Y_a(nj) = \sum_{i \text{ odd}}^{j} \binom{j}{i} X_a(n)^{j-i} Y_a(n)^i \left(\sqrt{d}\right)^{i-1}. \tag{2.22}$$

Hence

$$Y_a(nj) \equiv jX_a(n)^{j-1}Y_a(n) \left(\operatorname{mod} Y_a(n)^3\right). \tag{2.23}$$

For the proof of 2.20 in the $\Rightarrow$ direction, suppose $Y(n)^2|Y(m)$. Then by (2.19) $m = nj$ for some j. Let this be the j in (2.23). Since $X(n) \perp Y(n)$, (2.23) implies $Y(n)^2|jY(n)$. Hence $Y(n)|j$, so that $n \cdot Y(n)|m$. For the converse suppose $n \cdot Y(n)|m$. Let $j = Y(n)$. Then (2.23) $\Rightarrow$ $Y(n)^2|Y(n \cdot Y(n))$. So by (2.19), $Y(n)^2|Y(m)$.

LEMMA 2.24. *For $2 \leqslant a$ and $1 \leqslant n$,* $Y_a(n-1) + Y_a(n) < X_a(n)$.

Proof. Replace n by $n-1$ in (2.9) to obtain $a \cdot Y(n-1) + X(n-1) = Y(n)$. From $2 \leqslant a$, we get $2 \cdot Y(n-1) \leqslant a \cdot Y(n-1) < a \cdot Y(n-1) + X(n-1) = Y(n)$. Hence $Y(n-1) < Y(n) - Y(n-1)$. Add $Y(n)$ to both sides to obtain $Y(n-1) + Y(n) < 2 \cdot Y(n) - Y(n-1) \leqslant a \cdot Y(n) - Y(n-1) = X(n)$, by (2.10).

LEMMA 2.25. $Y_a(4ni \pm m) \equiv Y_a(m)$, $Y_a(4ni + 2n \pm m) \equiv \mp Y_a(m) \pmod{X_a(n)}$.

Proof. From (2.14) $Y(2n) \equiv 0 \pmod{X(n)}$. From (2.13) $X(2n) \equiv -1 \pmod{X(n)}$. So from (2.8) $Y(2n \pm m) \equiv Y(2n) \cdot X(m) \pm X(2n) \cdot Y(m) \equiv \mp Y(m) \pmod{X(n)}$, we have $Y(2n \pm m) \equiv \mp Y(m) \pmod{X(n)}$. Using this $Y(4n \pm m) = Y(2n + 2n \pm m) \equiv -Y(2n \pm m) \equiv \pm Y(m) \pmod{X(n)}$. So also $Y(4n \pm m) \equiv \pm Y(m) \pmod{X(n)}$. Here the signs ± correspond. In the next lemma they do not.

SECOND STEP DOWN LEMMA 2.26. $Y_a(k) \equiv \pm Y_a(m) \pmod{X_a(n)} \leftrightarrow k \equiv \pm m \pmod{2n}$.

Proof. We assume as usual $2 \leqslant a$, $1 \leqslant n$. For the proof in the $\Leftarrow$ direction, suppose that $k = 2nj \pm m$. When $j = 2i$, we have $Y(k) = Y(4ni \pm m) \equiv \pm Y(m) \pmod{X(n)}$. In the case that $j = 2i + 1$, we have $Y(k) = Y(4ni + 2n \pm m) \equiv \mp Y(m) \pmod{X(n)}$. So $Y(k) \equiv \pm Y(m) \pmod{X(n)}$.

For the proof in the $\Rightarrow$ direction, assume that $Y(k) \equiv \pm Y(m) \pmod{X(n)}$. Choose k' such that $0 \leqslant k' \leqslant n$ and $k \equiv \pm k' \pmod{2n}$. Choose m' such that $0 \leqslant m' \leqslant n$ and $m \equiv \pm m' \pmod{2n}$. Then from the assumption and Lemma 2.26 in the direction already proven, $\Leftarrow$, we have $Y(k') \equiv \pm Y(m') \pmod{X(n)}$. Hence it follows that $X(n) | Y(k') \pm Y(m')$. Therefore $k' = m'$, because if we suppose $k' \neq m'$, then we would have $0 < |Y(k') \pm Y(m')| \leqslant |Y(k') + Y(m')| \leqslant Y(n-1) + Y(n) < X(n)$, by Lemma 2.24. From $k' = m'$ we deduce $k \equiv \pm m \pmod{2n}$.

LEMMA 2.27. *For $A > 1$. In order that $C = Y_A(B)$, it is necessary and sufficient that there exist natural numbers D, E, F, G, H, I and i such that*

(1) $D^2 - (A^2 - 1)C^2 = 1$,	(4) $E = (i+1)2C^2$,	(7) $H \equiv C \pmod{F}$,
(2) $F^2 - (A^2 - 1)E^2 = 1$,	(5) $G \equiv A \pmod{F}$,	(8) $H \equiv B \pmod{2C}$,
(3) $I^2 - (G^2 - 1)H^2 = 1$,	(6) $G \equiv 1 \pmod{2C}$,	(9) $B \leqslant C$.

Proof of Sufficiency: Suppose there exist D, E, F, G, H, I, i satisfying (1)–(9). Then equations (1)–(3), Pell equations, imply the existence of numbers p, q, and r such that

$$D = X_A(p), \qquad C = Y_A(p), \qquad F = X_A(q),$$
$$E = Y_A(q), \qquad I = X_G(r), \qquad H = Y_G(r).$$

We also have $0 \leqslant p \leqslant C$ and $0 \leqslant B \leqslant C$. Hence the idea is to show $B = p$, by proving $B \equiv r \equiv \pm p \pmod{2C}$. We can suppose $0 < C$. Using the First Step Down Lemma, together with (4), we have

$$C^2 | E \Rightarrow Y_A^2(p) | Y_A(q) \Rightarrow Y_A(p) | q \Rightarrow C | q.$$

To show that $B \equiv r \pmod{2C}$ we use (6), (8) and the Congruence Rule (2.16) to get

$$B \equiv H = Y_G(r) \equiv Y_1(r) = r \pmod{2C} \ \Rightarrow\ B \equiv r \pmod{2C}. \tag{2.28}$$

By the Second Step Down Lemma, the Congruence Rule (2.15), by (5) and by (7),

$$Y_A(r) \equiv Y_G(r) = H \equiv C = Y_A(p) \left(\operatorname{mod} X_A(q)\right) \ \Rightarrow\ r \equiv \pm p \pmod{2q}.$$

But $C|q$. Hence $r \equiv \pm p \pmod{2C}$. This plus (2.28) implies $B \equiv \pm p \pmod{2C}$.

Proof of Necessity: Suppose $C = Y_A(B)$. Let $D = X_A(B)$. Then (1) and (9) hold. Put $q = B \cdot Y_A(B)$, $F = X_A(2q)$ and $E = Y_A(2q)$. Then (2) holds. Let $m = B \cdot Y_A(B)$ in the First Step Down Lemma (2.20). Then the First Step Down Lemma says

$$Y_A(B)^2 | Y_A(B \cdot Y_A(B)). \tag{2.29}$$

Hence $C^2 | Y_A(q)$. The Double Angle Formula (2.14) says $2X_A(q) \cdot Y_A(q) | Y_A(2q)$. Hence $2C^2 | E$. Therefore (4) can be satisfied. Put $G = A + F^2(F^2 - A)$. Then (5) holds. Also (2) and (4) together imply that $F^2 \equiv 1 \pmod{2C}$. Then $G = A + F^2(F^2 - A)$ implies (6). Put $I = X_G(B)$ and $H = Y_G(B)$. Then (3) holds. From (2.16) $H = Y_G(B) \equiv B \pmod{G - 1}$. So by (6) $H \equiv B \pmod{2C}$. Therefore (8) holds. From (2.15) we have $H = Y_G(B) \equiv Y_A(B) = C \pmod{G - A}$. This together with (5) implies $H \equiv C \pmod F$. Hence (7) holds.

§3. Exponential and Binomial Coefficient. By Lemma 2.27, the 3-place relation $y = Y_a(n)$ is diophantine. Hence we may use it in showing that other relations are diophantine. We can also use the 3-place relation $x = X_a(n)$ for this purpose. It is also diophantine. (That $x = X_a(n)$ is diophantine follows from Lemma 2.27 and equation (2.2).) We will use both $y = Y_a(n)$ and $x = X_a(n)$ in defining the exponential relation.

LEMMA 3.1. *Suppose* $1 \leqslant n$, $2 \leqslant k$. *For a sufficiently large*, $a \geqslant Y_k(n+1)$

$$k^n = rem\left(X_a(n) - (a - k)Y_a(n),\ \ 2ak - k^2 - 1\right).$$

Proof. From (2.17) we have $k \leqslant k^n < (2k-1)^n \leqslant Y_k(n+1) \leqslant a$. From this it follows that $k + 1 \leqslant a$, and hence $a < a \cdot k < a \cdot k + k - 1 = ak + (k+1)k - k^2 - 1 \leqslant ak + ak - k^2 - 1 = 2ak - k^2 - 1$. Therefore $k^n < 2ak - k^2 - 1$, k^n is smaller than the modulus. But by congruence (2.18), $k^n \equiv X_a(n) - (a - k)Y_a(n) \pmod{2ak - k^2 - 1}$. Since k^n is less than the modulus, it is the remainder.

Lemmas 2.27, 3.1 and (1.7) imply that the exponential relation, $m = k^n$ is diophantine. By Lemma 3.1, $m = k^n$ if and only if there exists an a such that

$$\begin{aligned} &m \equiv X_a(n) - (a - k) \cdot Y_a(n) \pmod{2ak - k^2 - 1}, \\ &m < a \qquad a \geqslant Y_k(n+1). \end{aligned} \tag{3.1}$$

We show next that the binomial coefficient is diophantine. This relation was first proved exponential diophantine by Julia Robinson [1952]. The basic idea of the proof is that the binomial coefficients are just the digits in the base u expansion of $(1 + u)^n$, when u is large enough.

LEMMA 3.2. *For* $0 \leqslant k$, *n and* $u > 2^n$,

$$\binom{n}{k} = rem\left(\left\lfloor \frac{(u+1)^n}{u^k} \right\rfloor, u\right).$$

Proof. Expand $(u+1)^n$ by the Binomial Theorem and divide by u^k to get,

$$\frac{(u+1)^n}{u^k} = \sum_{i=k+1}^{n} \binom{n}{i} u^{i-k} + \binom{n}{k} + \sum_{i=0}^{k-1} \binom{n}{i} u^{i-k}. \tag{3.2}$$

From (3.2) together with the inequality $u^{i-k} \leqslant 1/u$, which holds for $i \leqslant k-1$, we get

$$\sum_{i=0}^{k-1} \binom{n}{i} u^{i-k} \leqslant \sum_{i=0}^{k-1} \binom{n}{i} \frac{1}{u} \leqslant \frac{1}{u} \sum_{i=0}^{n} \binom{n}{i} = \frac{2^n}{u} < 1. \tag{3.3}$$

Hence

$$\left\lfloor \frac{(u+1)^n}{u^k} \right\rfloor \equiv \binom{n}{k} \pmod{u}.$$

So Lemma 3.2 follows from $\binom{n}{k} \leqslant 2^n < u$.

Lemmas 2.27, 3.1, 3.2, 1.7 and 1.8 imply that the binomial coefficient is diophantine. We have $m = \binom{n}{k}$ iff there exist u, x, and y, such that

$$(u+1)^n = y \cdot u^{k+1} + m \cdot u^k + x, \qquad 2^n < u, \qquad x < u^k \qquad \text{and} \qquad m < u. \tag{3.4}$$

We now use this to prove that the following relation $\preccurlyeq$ (*bit masking*) is diophantine.

DEFINITION 3.5. *Let r and s be natural numbers written in binary,* (*base* 2). *Then* $r \preccurlyeq s$ *means each binary digit of r is less than or equal to the corresponding binary digit of s.*

The masking relation $\preccurlyeq$ is closely related to the operation of taking the logical *and* of two numbers. One can define $\preccurlyeq$ from logical and by

$$a \preccurlyeq b \Leftrightarrow a \wedge b = a. \tag{3.6}$$

Logical and is definable from $\preccurlyeq$ by

$$a \wedge b = c \;\Leftrightarrow\; c \preccurlyeq a \qquad \text{and} \qquad a \preccurlyeq a + b - c. \tag{3.7}$$

Another property of the masking relation is that $r \preccurlyeq s$ implies $r \leqslant s$. The masking relation also satisfies the axioms for a partial ordering, reflexivity, anti-symmetry and transitivity.

The masking relation $\preccurlyeq$ is also diophantine. For the proof we will use the Theorem of E. Lucas [1878] on binomial coefficients modulo a prime.

THEOREM 3.8 (Lucas [1878]). *If p is prime,* r_i, s_i *the base p digits of r and s, then*

$$\binom{s}{r} = \binom{s_n}{r_n} \cdots \binom{s_1}{r_1}\binom{s_0}{r_0} \pmod{p}. \tag{3.8}$$

Proof. It is easy to see, by induction on n, that this general form of Lucas Theorem, Theorem 3.8, is implied by the following special case (H. Anton [1869], approximately $n = 1$), of Lucas' Theorem:

LEMMA 3.9. *Let p be a prime. Suppose $0 \leqslant b < p$ and $0 \leqslant d < p$. Then*

$$\binom{a \cdot p + b}{c \cdot p + d} \equiv \binom{a}{c} \cdot \binom{b}{d} \pmod{p}. \tag{3.9}$$

Proof. Working in the polynomial ring $\mathbb{Z}_p[x]$, consider the coefficient of the term x^{cp+d} in the polynomial $(1+x)^{ap+b}$. From the equality $(x+y)^p = x^p + y^p$ we get

$$\sum_{n=0}^{ap+b} \binom{ap+b}{i} x^n = (1+x)^{a \cdot p+b} = (1+x)^{pa}(1+x)^b = (1+x^p)^a \cdot (1+x)^b$$

$$= \left(\sum_{j=0}^{a} \binom{a}{j} x^{jp}\right)\left(\sum_{i=0}^{b} \binom{b}{i} x^i\right) = \sum_{j=0}^{a} \sum_{i=0}^{b} \binom{a}{j}\binom{b}{i} x^{jp+i}.$$

The coefficient of x^{cp+d} must be the same in both polynomials. However because of the inequalities $0 \leqslant i \leqslant b < p$ and $0 \leqslant d < p$, the equation $jp + i = cp + d$ can hold only when $i = d$ and $j = c$. Therefore, the coefficient of the term x^{cp+d} in $(1+x)^{ap+b}$ must therefore be $\binom{a}{c}\binom{b}{d}$. Hence $\binom{ap+b}{cp+d} = \binom{a}{c} \cdot \binom{b}{d}$ holds in $\mathbb{Z}_p[x]$.

Now we can prove that the masking relation $\preccurlyeq$ is diophantine. This will follow from

LEMMA 3.10.

$$r \preccurlyeq s \Leftrightarrow \binom{s}{r} \equiv 1 \pmod{2}.$$

Proof. Put $p = 2$ in Theorem 3.8. Then 3.10 follows from the trivial relations

$$\binom{1}{0} = 1, \quad \binom{1}{1} = 1, \quad \binom{0}{0} = 1, \quad \binom{0}{1} = 0.$$

Lemma 3.10 is proved. From this Lemma together with Lemma 3.10, 3.2, (1.7) and (1.8), it follows that $\preccurlyeq$ is a diophantine relation.

§4. Arithmetization of register machines. We now have enough diophantine relations to prove that every recursively enumerable set is diophantine. Recall the definition of r.e. set: A set A is r.e. if and only if A is empty or the range of a recursive function. For the definition of recursive function, we will use the register machine, a model of computation equivalent to the Turing machine. A *register machine* is a "machine" with a finite program and a finite number of separately addressable registers, $R1, R2, \ldots, Rr$. The registers are assumed unbounded, each register can contain an arbitrarily large nonnegative integer. A subset of the registers, say $R_1, R_2, \ldots, R_k$, $(k < r)$, is designated as *input* registers and a subset, say $R_1, R_2, \ldots, R_m$, $(m < r)$, as *output* registers. This is to handle functions of k variables whose values may be m-tuples. For a function of one variable of course $k = 1$. Normally also $m = 1$. Usually R_1 is thought of as the input–output register.

A register machine is actually a program, a list of commands written on lines labelled $L1, L2, \ldots, Ll$. The register machine's commands are normally executed in sequence; however, the register machine may execute commands which tell it to transfer to a different location and begin execution there.

For the computation of all recursive functions, it is sufficient to assume that the register machine is capable of adding or subtracting 1 from a register and of transferring on zero.

The subtraction command can cause a problem if one attempts to subtract 1 from an already zero register. For this reason Minsky [1967] permitted the subtraction command to occur only after a test for zero. His subtraction command therefore required two lines

$$Li \qquad \text{IF } Rj = 0, \text{GOTO } Lk, \tag{4.1}$$

$$Li + 1 \quad \text{ELSE } Rj \leftarrow R_j - 1. \tag{4.2}$$

It is sufficient to assume that the program is written in such a way that subtraction from a zero register never occurs. We will assume that our machine has the following one line commands:

	COMMAND	INTERPRETATION	
Li	GOTO *Lk*	*Transfer to line Lk.*	(4.3)
Li	IF 0 < *Rj* GOTO *Lk*	*Conditional transfer to line Lk.*	(4.4)
Li	$Rj \leftarrow Rj + 1$,	*Increment register Rj.*	(4.5)
Li	$Rj \leftarrow Rj - 1$,	*Decrement register Rj.*	(4.6)

In fact because it shortens our example and does not complicate the proof, we will allow several commands of type (4.5) and (4.6) to be written on one line, in parallel, when they refer to different registers. Below is an example of a register machine.

EXAMPLE 1. *A register machine which computes the x^{th} Fibonacci number, F_x.*

```
L1    IF R1 = 0,   GOTO L20
L2    R2 ← R2 + 1, R3 ← R3 + 1
L3    R1 ← R1 − 1
L4    IF R1 = 0,   GOTO L16
L5    R1 ← R1 − 1
L6    R4 ← R4 + 1, R5 ← R5 + 1
L7    R3 ← R3 − 1
L8    IF 0 < R3,   GOTO L6
L9    R4 ← R4 + 1, R2 ← R2 − 1
L10   IF 0 < R2,   GOTO L9
L11   R3 ← R3 + 1, R4 ← R4 − 1
L12   IF 0 < R4,   GOTO L11
L13   R2 ← R2 + 1, R5 ← R5 − 1
L14   IF 0 < R5,   GOTO L13
L15   IF 0 < R1,   GOTO L5
L16   R3 ← R3 − 1
L17   IF 0 < R3,   GOTO L16
L18   R2 ← R2 − 1, R1 ← R1 + 1,
L19   IF 0 < R2,   GOTO L18,
```

A function is recursive (computable) if and only if it can be computed by a register machine (Minsky [1961] [1967], Melzak [1961], Lambek [1961], Shepherdson and Sturgis [1963].) The machine M of Example 1 computes the function $f(x) = F_x$, where F_x is the xth Fibonacci number, (Fibonacci number with index x in the enumeration $F_0 = 0$, $F_1 = 1$, $F_{x+2} = F_x + F_{x+1}$). If M is started on line $L1$ with x in register $R1$ and zeros in the other registers, then M eventually halts with $f(x) = F_x$ in register $R1$ and zeros in the other registers. We can understand this as a general definition of computability.

To arithmetize the work of an arbitrary register machine, we will use digits of numbers written to a base Q, where Q is power of 2. Consider the register machine M of Example 1 or more generally any register machine with r registers and l lines in its program. Assume that M computes the total function $y = f(x)$. During the computation the contents of a register at time t can never exceed $x + t$. We will use the fact that after s steps in the computation, the contents of each register Ri is $\leqslant x + s$.

Suppose that after s steps the value $y = f(x)$ is obtained by M. Let $r_{j,t}$ denote the *contents* of register Rj at time t during the course of the computation. If Q is large enough, we will have $r_{j,t} < Q$. So the numbers $r_{j,t}$ may be considered to be the digits of a number written to the base Q. But we will need more room. We will require $2r_{j,t} < Q$, (for the reason see the explanation following (4.19)). Hence we will require that Q satisfy:

$$x + s < Q/2, \tag{4.7}$$

$$l + 1 < Q, \tag{4.8}$$

$$Q \operatorname{pow} 2. \tag{4.9}$$

For example the number $Q = 2^{x+s+l+1}$ will be large enough. To describe the location of the machine in the code at each time, let $l_{j,t}$ be 1 or 0 according as we execute, or do not execute the instruction(s) at location Lj at time t. When the machine M computes, it will then generate numbers R_j and L_i where

$$R_j = \sum_{t=0}^{s} r_{j,t} Q^t \qquad \left(0 \leqslant r_{j,t} < \frac{Q}{2}\right), \tag{4.10}$$

$$L_i = \sum_{t=0}^{s} l_{i,t} Q^t \qquad (0 \leqslant l_{i,t} \leqslant 1). \tag{4.11}$$

Here the index j runs from 1 to r, where r is the number of *registers*. The index i runs from 1 to l, where l is the number of *lines* in the program.

It will be useful also to have a number I which when written to the base Q has all digits equal to 1. The geometric series will allow us to obtain such a number I.

$$\text{If} \quad 1 + (Q-1)I = Q^{s+1}, \quad \text{then} \quad I = \sum_{t=0}^{s} Q^t. \tag{4.12}$$

To visualize the arithmetization, consider the machine of Example 1 which has $r = 5$ registers and $l = 19$ lines in its program. When it is started on $x = 2$ as input (i.e. with $x = 2$ in register $R1$), the machine will be seen to halt after $s = 23$ steps, with 1 in register $R1$ and zeros in the other registers. (This is because $F_2 = 1$ is the second Fibonacci number in the standard enumeration $0, 1, 1, 2, 3, 5, 8, 13, \ldots$.) During the computation of F_2, the machine will generate the following numbers R_1, R_2, R_3, R_4, R_5, representing successive contents of the registers, in the sense of (4.10). These numbers, when written in base Q, would look as follows:

$$R_1 = 110000000000000000011222,$$
$$R_2 = 001111111000000111111100,$$
$$R_3 = 000011222221100001111100,$$
$$R_4 = 000000000001122111000000,$$
$$R_5 = 000000000111111111000000.$$

Similarly, numbers representing the sequence of program locations $L_1, L_2, \ldots, L_{20}$, will be generated during the computation of F_2. In base Q they will look as follows:

$$L_1 = 000000000000000000000001,$$
$$L_2 = 000000000000000000000010,$$
$$L_3 = 000000000000000000000100,$$
$$L_4 = 000000000000000000001000,$$
$$L_5 = 000000000000000000010000,$$
$$L_6 = 000000000000000000100000,$$
$$L_7 = 000000000000000001000000,$$
$$L_8 = 000000000000000010000000,$$
$$L_9 = 000000000000000100000000,$$
$$L_{10} = 000000000000001000000000,$$
$$L_{11} = 000000000001010000000000,$$
$$L_{12} = 000000000010100000000000,$$
$$L_{13} = 000000000100000000000000,$$
$$L_{14} = 000000001000000000000000,$$
$$L_{15} = 000000010000000000000000,$$
$$L_{16} = 000010100000000000000000,$$
$$L_{17} = 000101000000000000000000,$$
$$L_{18} = 001000000000000000000000,$$
$$L_{19} = 010000000000000000000000,$$
$$L_{20} = 100000000000000000000000,$$

Notice that the generated numbers $L_1 L_2, \ldots, L_{20}, R_2, R_3, R_4, R_5$ contain the entire history of the computation of $f(2) = F_2 = 1$.

Now we show how to write down diophantine conditions on arbitrary variables $L_1, L_2, \ldots, L_{20}, R_1, R_2, R_3, R_4, R_5, s, Q, x, y$ sufficient to force them to be these particular values. That is, we give Diophantine conditions on these unknowns which are satisfiable if and only if, on input x, the machine M produces the output value $y = f(x)$. The unknowns in these diophantine relations will be s, Q, $I, R_1, \ldots, R_r, L_1, \ldots, L_{1+1}$. The quantities r and l will be constants, (e.g. $r = 5$ and $l = 19$ in the example). After the conditions have been written down, the methods of §1, §2 and §3 may be used to translate them into diophantine equations in these same unknowns, and more unknowns. The variables x and y will be parameters in the resulting equations.

The first four conditions will be (4.7), (4.8), (4.9) and (4.12). Then to force an arbitrary natural number R_j to have the form (4.10), we will use the masking relation

$$R_j \preccurlyeq \left(\frac{Q}{2} - 1\right) \cdot I, \qquad (j = 1, \dots, r). \tag{4.13}$$

Condition (4.13) is implied by condition (4.10), because Q is a power of 2. We also need to force exactly one digit to be 1 in every column of the L matrix. Since $l < Q$ by (4.8), we can use for this purpose the following two conditions:

$$I = \sum_{i=1}^{l+1} L_i, \tag{4.14}$$

$$L_i \preccurlyeq I \qquad (i = 1, \dots, l+1). \tag{4.15}$$

To start the machine on line $L1$, we stipulate as a starting condition that

$$1 \preccurlyeq L_1. \tag{4.16}$$

By means of GOTO instructions we can suppose there is only one stop command, and that it is located at the end of the program, on line L_{l+1}. Then the condition for stopping the machine after s steps can simply be

$$L_{l+1} = Q^s. \tag{4.17}$$

To simulate each GOTO command (unconditional transfer), Li GOTO Lk, we include a condition

$$QL_i \preccurlyeq L_k. \tag{4.18}$$

This forces the $t + 1$st digit of L_k to be 1 whenever the tth digit of L_i is 1. Here $1 \leqslant k \leqslant l + 1$. (Line $Ll + 1$ is permitted to be the target of a GOTO.) This same idea, (4.18) can be used also for the conditional transfer command (4.4), Li IF $0 < Rj$, GOTO Lk. (Assume $k \neq i + 1$ so the command is not trivial.) Here we use two masking conditions:

$$QL_i \preccurlyeq L_k + L_{i+1} \qquad \text{and} \qquad QL_i \preccurlyeq L_k + QI - 2R_j. \tag{4.19}$$

The first masking condition forces transfer toline Li or line $Li + 1$. The second decides which one it will be. A diagram explains how this works:

```
                Q^6          Q^5          Q^4          Q^3          Q^2          Q^1          Q^0
      I =0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1.
    Q·I =0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0
  2·R_j =0 0 0 0 0 0 * * * * * 0 * * * * * 0 * * * * * 0 * * * * * 0 * * * * * 0 * * * * * 0.
```

Here $Q = 2^6$ and $s = 5$ and the idea is that subtraction of $2R_j$ from QI pulls a bit from the Q^{t+1} position of $Q \cdot I$ when $0 < r_{j,t}$. Thus (4.19) implies that $0 < r_{j,t}$ if and only if $l_{k,t+1} = 1$. The reason why we needed $2r_{j,t} < Q$ and why we divided Q by 2 in (4.13) can now be understood. The purpose was to create the zeros between adjacent blocks of digits in $2R_j$. The reason we use $QI - 2R_j$ and not $I - 2R_j$ is that we could not be sure that $I - 2R_j$ would be nonnegative.

The command Li IF $Rj = 0$, GOTO Lk may be treated analogously. This command is just the opposite of (4.4), so it can be simulated with a masking condition like (4.19) but with the second occurrence of L_k replaced by L_{i+1}. This has the effect of causing the machine to go to $Li + 1$ instead of Lk, when $Rj > 0$. (Again we are supposing that $k \neq i + 1$ so that Li IF $Rj = 0$, GOTO Lk is nontrivial.)

While commands of type $Li\ Rj \leftarrow Rj \pm 1$ are not usually thought of as transfer commands, there is an implied GOTO associated with them; GOTO the next line. So, for each occurrence of such a command (more precisely for each line Li containing occurrences of such commands), we need one corresponding masking condition:

$$QL_i \preccurlyeq L_{i+1}. \tag{4.20}$$

Finally it is necessary to include for each register an equation which ensures that the contents of the register is equal to the tth Q-ary digit of the corresponding number Rj at each time t. We call these *register equations*. For the example program they would be:

$$R_1 + yQ^{s+1} = QR_1 + x + \sum_k QL_k - \sum_i QL_i, \tag{4.21}$$

$$R_j = QR_j + \sum_k QL_k - \sum_i QL_i, \quad (\text{for } j = 2, 3, \dots, r). \tag{4.22}$$

The k-sums are over all k for which there is an instruction of type $Lk\ Rj \leftarrow Rj + 1$, on line Lk. The i-sums are over all i for which the program has a command of type $Li\ Rj \leftarrow Rj - 1$ on line Li.

The register equation for register $R1$ is different from the register equations for the other registers because $R1$ is the input–output register (assuming functions of 1 variable). At time $t = 0$, register R_1 contains x. At time s it contains y. The other registers contain zeros at these times.

If more registers are designated as input–output registers, for example if $m > 1$, then it is necessary to add a term x to the right side of (4.22) and a term yQ^{s+1} to the left side of (4.22), for the corresponding register equation.

From these equations and conditions, it is not difficult to show, by induction on t_1, that if machine M computes function f, then for all x, y, conditions (4.7), (4.8), (4.9), (4.12)–(4.22) have a solution in the unknowns $s, Q, I, R_1, \dots, R_r, L_1, \dots, L_{l+l}$ if and only if $f(x) = y$. Methods of §1, §2, and §3 then permit us to write conditions (4.7), (4.8), (4.9), (4.12)–(4.22) as polynomial equations in the unknowns $s, Q, I, R_1, \dots, R_r, L_1, \dots, L_l$, and more unknowns. After we rename the unknowns $x_1, x_2, \dots, x_n$, transpose all the terms in the equations to one side and sum squares as in (1.5), we obtain a single polynomial equation $P(x, y, x_1, x_2, \dots, x_n) = 0$ with the property that $(\exists x_1, x_2, \dots, x_n)$ $[P(x, y, x_1, x_2, \dots, x_n) = 0]$ if and only if $f(x) = y$.

To complete the proof of Theorem 1, let A be any r.e. set. Suppose $A \neq \emptyset$. Let f be a recursive function whose range is A, so that $y \in A$ iff $(\exists x)[f(x) = y]$. Let M be a register machine computing f. When A is a 1-ary relation, $A(a_1)$, we have $a_1 \in A$ iff $(\exists x)$ $[f(x) = a_1]$. Thus $a_1 \in A$ if and only if $(\exists x, x_1, x_2, \dots, x_n)$ $[P(x, a_1, x_1, x_2, \dots, x_n) = 0]$. When A is an m-ary relation, $A(a_1, a_2, \dots, a_m)$, then we have $(a_1, a_2, \dots, a_m) \in A$ iff $(\exists x)$ $[f(x) = (a_1, a_2, \dots, a_m)]$. So P must include m copies of register equation (4.21), with yQ^{s+1} replaced by a_iQ^{s+1} $(i = 1, \dots, m)$. Then $A(a_1, a_2, \dots, a_m)$ holds iff $(\exists x, x_1, \dots, x_n)$ $[P(x, a_1, \dots, a_m, x_1, \dots, x_n) = 0]$. Theorem 1 is proved.

§5. Hilbert's 10th problem is unsolvable. We give here a proof that Theorem 1 implies the recursive unsolvability of Hilbert's Tenth Problem. To shorten the proof T. Rado's game [1962] will be used.

THEOREM 2. *There exists no algorithm to solve Hilbert's Tenth Problem.*

Proof. Consider the problem of programming a register machine to return the largest possible number in register $R1$ and stop. Suppose the registers are initially set to 0. Let $R(l)$ be the largest possible number generable in this way, by an l line register machine with registers set initially to 0.

Restricting subtraction commands to the (safe) type (4.1)–(4.2), and disallowing parallelization of commands, $R(l)$ is evidently a well defined function of l. A finite set has a greatest element. There are only a finite number of l line register machines.

Evidently we have $R(1) = 1$ and $R(2) = 2$. Also R is an increasing function of l, $R(l) < R(l+1)$. This can be seen by adding a new line, $R1 \leftarrow R1 + 1$, to the end of an l line program.

Define a binary relation S by putting $S = \{(k,l)\colon k \leqslant R(l)\}$. Then

$$(k,l) \in S \Leftrightarrow (\exists s)[\text{some } l \text{ line machine halts in } s \text{ steps with contents of } R1 \geqslant k]. \tag{5.1}$$

From (5.1) it follows that the set S is r.e. This set S is related to the function R by

$$R(l) = \min k[\neg S(k+1,l)]. \tag{5.2}$$

Apply Theorem 1 to S. The result is a polynomial $P(k,l,x_1,\dots,x_n)$ with the property that

$$S(k,l) \Leftrightarrow (\exists x_1,x_2,\dots,x_n)[P(k,l,x_1,x_2,\dots,x_n) = 0]. \tag{5.3}$$

From (5.2) and (5.3) it follows that

$$R(l) = \min k[\neg(\exists x_1,x_2,\dots,x_n)[P(k+1,l,x_1,x_2,\dots,x_n) = 0]]. \tag{5.4}$$

Hence if Hilbert's Tenth Problem were solvable, then R would be computable. But the function R grows very rapidly. In fact it is noncomputable.

LEMMA 5.5. *Let f be any computable function. Then for l sufficiently large, $f(l) < R(l)$.*

Proof. Without loss of generality we can suppose that f is an increasing function. Let N be a register machine which computes the function f. Suppose N has c lines in its program. Let F be the machine obtained by adding the instruction $R1 \leftarrow R1 + 1$ to the end of N. Then F has $c+1$ lines and it computes the function $f(x) + 1$.

Let D be a 5 line register machine with the property that when started with x in $R2$ places $2x$ in $R1$ and stops. D could be the machine,

```
L1   IF $R2 = 0$, GOTO $L6$,
     $R2 \leftarrow R2 - 1$,
     $R1 \leftarrow R1 + 1$,
     $R1 \leftarrow R1 + 1$,
     GOTO $L1$,                          (5.6)
```

For each x, let M_x be a register machine with x lines and the property that when started with 0 in $R2$ produces x in $R2$. For example M_x could be taken to be a straight line program consisting of x copies of $R2 \leftarrow R2 + 1$.

Now consider the register machine $F(D(M_x))$. By the notation $F(D(M_x))$ is meant M_x followed by D followed by F, in that order. $F(D(M_x))$ is a register machine with $x + 6 + c$ lines in its program. It has the property that for each x, when started with registers initially set to zero, it will produce the number $f(2x) + 1$ in $R1$, then halt. The existence of this machine proves that

$$f(2x) < R(x + 6 + c) \tag{5.7}$$

for each x. When $x \geqslant 6 + c$, we have $x + 6 + c \leqslant 2x$. Hence

$$f(x + 6 + c) \leqslant f(2x), \tag{5.8}$$

because f is increasing. Now put $l = x + 6 + c$. Then for $l \geqslant 12 + 2c$, (5.7) and (5.8) imply $f(l) < R(l)$. Lemma 5.5 is proved. Hilbert's Tenth Problem is unsolvable.

§6. Every computable function is a polynomial. Unsolvability is a negative result, but the solution of Hilbert's Tenth Problem has many positive consequences. One is the existence of polynomials whose set of positive values coincides with any given r.e. set. This result is well known, so we shall prove a stronger form of it which at first glance appears even more surprising.

THEOREM 3. *Let f be any computable function. Then there exists a polynomial Q, with integer coefficients, such that for all nonnegative integers x and y*

$$f(x) = y \leftrightarrow (\exists x_0, x_1, \dots, x_n)[Q(x, x_0, x_1, \dots, x_n) = y]. \tag{6.1}$$

Proof. We apply Theorem 1 to the graph of f (which is an r.e. set), and then use a trick due to H. Putnam [1960]. Starting from the polynomial P which Diophantine defines the graph of f, and using also the fact that f is nonnegative, $(\forall x)[f(x) \geqslant 0]$, we get

$$\begin{aligned} f(x) = y &\Leftrightarrow (\exists x_1, \dots, x_n)(P(x, y, x_1, \dots, x_n) = 0), \\ &\Leftrightarrow (\exists x_0, \dots, x_n)\big(1 - P(x, x_0, x_1, \dots, x_n)^2 > 0 \text{ and } x_0 = y\big), \\ &\Leftrightarrow (\exists x_0, \dots, x_n)\Big((x_0 + 1)\big[1 - P(x, x_0, x_1, \dots, x_n)^2\big] = y + 1\Big). \end{aligned}$$

In the $\Leftarrow$ direction we are using the fact that $1 - P(x, x_0, x_1, \dots, x_n)^2 > 0$ implies $P(x, x_0, x_1, \dots, x_n) = 0$ which implies $f(x) = x_0$. So we can put $Q(x, x_0, \dots, x_n) = (x_0 + 1)[1 - P(x, x_0, x_1, \dots, x_n)^2] - 1$.

To illustrate Theorem 3, we could apply it to the function $f(x) = F_x$, whose range is the set of Fibonacci numbers. In this case the theorem would give us a polynomial $Q(x, x_0, \dots, x_n)$ such that for all x, y

$$F_x = y \Leftrightarrow (\exists x_0, \dots, x_n)[Q(x, x_0, \dots, x_n) = y]. \tag{6.3}$$

This would be a Fibonacci number representing polynomial which gives the sequence in order, as a function of x.

The proof of Theorem 1 shows how to construct such a polynomial. If we use this proof, the number n of unknowns will be rather excessive. By another construction, however, it can be shown that we need only nine unknowns. $n = 9$ here and generally in Theorems 1 and 3. This result, the nine unknowns theorem, is a theorem of the second author. A proof can be found in a paper of the first author [1982].

REFERENCES

[1869] H. Anton, Arch. Math. Phys. Vol. 49 (1869), pp. 303–306. (L. E. Dickson, History of the Theory of Numbers, vol. 1, Carnegie Institute of Washington, 1919, 1920, 1923. Chelsea Publishing Co., N.Y., 1971, p. 271.)

[1970] G. V. Chudnovskii, Diophantine predicates (Russian), *Uspekhi Matematicheskikh Nauk*, 25 (1970), 185–186.

[1958] M. D. Davis, Computability and Unsolvability, McGraw-Hill, New York, 1958, xxv + 210 pp.

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

[1971] M. D. Davis, An explicit diophantine definition of the exponential function, *Comm. Pure and Applied Math.*, 24 (1971), 137–145.

[1973] ______, Hilbert's tenth problem is unsolvable, this MONTHLY, 80 (1973), 233–269. MR 47, #6465.

[1976] M. D. Davis, Y. V. Matijasevič & J. Robinson, Hilbert's tenth problem, diophantine equations: positive aspects of a negative solution, Mathematical Developments Arising from the Hilbert problems, Proc. of Symposia in Pure Math., 28, American Math. Soc., Providence, RI, 1976, 323–378. MR55, #5522.

[1980] C. Dimitracopoulos, Matijasevič's Theorem and Fragments of Arithmetic, Dissertation, University of Manchester, 1980.

[1900] D. Hilbert, Mathematische Problem, Vortrag, gehalten auf dem Internationaler Mathematiker Kongress zu Paris 1900, Nachrichten der Akademie der Wissenschaften in Göttingen.II. Mathematische-Physikalische Klasse, 1900, 253–297. English transl: *Bulletin of the American Mathematical Society* 8 (1901–1902), 437–479.

[1974] J. P. Jones, Recursive undecidability—An exposition, MONTHLY 81 (1974), pp. 724–738.

[1976] J. P. Jones, D. Sato, H. Wada and D. Wiens, Diophantine Representation of the Set of Prime Numbers, MONTHLY 83 (1976) 449–464. MR 54, 2615.

[1975] J. P. Jones, Diophantine Representation of the Fibonacci Numbers, *Fibonacci Quarterly*, 13, (1975), 84–88. MR 52, 3035.

[1989] ______, Diophantine Representation of the set of Fibonacci Numbers Over Natural Numbers, Proceedings of the Third International Conference on Fibonacci Numbers and Their Applications, University of Pisa, July 25–29, 1988, Pisa, Italy. Kluwer Academic Publishers.

[1982] ______, Universal diophantine equation, *J. of Symbolic Logic*, 47 (1982), 549–571. MR 84e, #10070.

[1982] J. P. Jones and Y. V. Matijasevič, 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, 107, North-Holland, Amsterdam, 1982, 159–177. MR 85i, #03138.

[1984] ______, Register machine proof of the theorem on exponential diophantine representation of enumerable sets, *J. of Symbolic Logic*, 49 (1984), 818–829. MR 85i, #03139.

[1971] N. K. Kosovskii, On diophantine representation of the sequence of solutions of Pell's equation (Russian), Zapiski Nauchnyh Seminarov Leningradskogo Otdeleniya Matematicheskogo Instituta im. V. A. Steklova Akademia Nauk SSSR, 20 (1971), 49–59. English translation: *J. of Soviet Mathematics*, 1 (1973), 28–35.

[1852] E. E. Kummer, Über die Ergänzungssatze zu den Allgemeinen Reziprozitatsgesetzen, *J. für die Reine und Angewandte Mathematik*, 44 (1852), 93–146.

[1770] J. L. Lagrange, Nouv. Mem. Acad. Roy. Sc. de Berlin, 1770, Berlin, 1772, 123–133; *Oeuvres*, 3, 1869, 189–201. See also the edition of Diophantus published by G. Wertheim, 324–330.

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

[1930] D. H. Lehmer, An extended theory of Lucas' functions, *Annals of Math* 31 (1930), 419–448.

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

[1878] ______, Theorie des fonctions numériques simplement périodiques, *American Journal of Mathematics*, 1 (1878), 184–240, 289–321. English translation: The Fibonacci Association, Santa Clara University, CA 95053 (1969).

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

[1971] ______, Diophantine representation of enumerable predicates, *Izvestija Akademii Nauk SSSR Serija Mathematiceskaja*, 35 (1971), 3–30; English Transl. *Mathematics of the USSR–Izvestija*, 5 (1971), 1–28. MR 43, #54.

[1972] ______, Diophantine sets, *Uspekhi Matematičeskih Nauk*, 27 (1972), no. 5 (167), 185–222. English translation, *Russian Mathematical Surveys*, 27 (1972), No. 5, 124–164. MR 56, #109.

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

[1984] Y. V. Matijasevič, On investigations of some algorithmic problems in algebra and number theory, in Algebra, Math. Logic, Theory of Numbers and Topology, 50 years of M.I.A.N., Trudy Mathematicheskogo Instituta Steklova, Nauka, 168 (1984), pp. 218–235. English translation: American Mathematical Society, Proceedings of the Steklov Institute (1986), No. 3, pp. 227–252.

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

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

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

[1971] R. Parikh, Existence and feasibility in arithmetic, *J. Symbolic Logic*, 36 (1971), 494–508.

[1982] J. B. Paris and C. Dimitracopoulos, Truth definitions for Δ_0 formulae, Logic and Arithmetc, Monographie No. 30, L'Enseignement Mathematique, Engeler, Lauchli & Strassen eds., University of Geneva, 317–330.

[1960] H. Putnam, An unsolvable problem in number theory, *J. of Symbolic Logic* 25 (1960), 220–232.

[1962] T. Rado, On noncomputable functions, *Bell Sys. Tech. Jour.*, 41 (1962) 877–884.

[1952] J. Robinson, Existential definability in arithmetic, *Trans. Amer. Math. Soc.*, 72 (1952), 437–449. MR 14, #4.

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

[1967] H. Rogers Jr., Theory of Recursive Functions and Effective Computability, McGraw-Hill, New York, 1967, ix + 482 pp.

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