2. PAIRS AND SEQUENCES

Cantor proved that the set of ordered pairs of natural numbers can be put in one-to-one correspondence with the set of natural numbers. This can be done in many ways, of course, but there is one particularly simple mapping function which we shall find useful. Let $J(x, y) = \frac{1}{2}((x + y)^2 + 3x + y)$. Then $J(x, y)$ is an integer-valued function which assumes each natural number as value exactly once for x and y natural numbers. In fact, $J(0, 0) = 0$ and

$$J(x, y) + 1 = \begin{cases} J(x + 1, y - 1), & \text{if } y > 0, \\ J(0, x + 1), & \text{if } y = 0. \end{cases}$$

So J maps the set of ordered pairs of natural numbers onto the natural numbers according to the following diagram.

	$J(x,y)$	y=0	1	2	3	4
	0	0	1	3	6	10
	1	2	4	7	11	
x	2	5	8	12		
	3	9	13			
	4	14				

Two inverse functions K and L are uniquely determined by the equation $u = J(Ku, Lu)$.† The first few values of K and L which clearly indicate the general pattern of their sequence of values are given below:

u	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Ku	0	0	1	0	1	2	0	1	2	3	0	1	2	3	4	0
Lu	0	1	0	2	1	0	3	2	1	0	4	3	2	1	0	5

† We usually omit parentheses in the functional notation for functions of one variable. Thus $F(x)$ is written Fx and $F(G(x))$ is written FGx.

To give an example of the use of pairing functions, suppose $\mathfrak{S}$ is the range of a function $F(x, y)$ of two variables. Then $\mathfrak{S}$ is also the range of $G(u) = F(Ku, Lu)$, a function of one variable. We can also recover the function F from G, since $F(x, y) = G(J(x, y))$.

We can also let a set $\mathfrak{S}$ of natural numbers represent a binary relation $\mathfrak{R}$ by the equivalence: $\mathfrak{R}(x, y)$ if and only if $J(x, y) \in \mathfrak{S}$. Thus, a sequence of sets $\mathfrak{S}_0, \mathfrak{S}_1, \mathfrak{S}_2, \cdots$ is represented by a single set $\mathfrak{S}$ through the correspondence $x \in \mathfrak{S}_n$ if and only if $J(n, x) \in \mathfrak{S}$.

Another example of the usefulness of pairing functions is a simple system of numbering diophantine equations. Here a diophantine equation is an equation of the form $F(x_0, x_1, \cdots) = G(x_0, x_1, \cdots)$ where F and G are terms built up from $x_0, x_1, \cdots$ and natural numbers by addition and multiplication. A diophantine equation can also be put in the form $P(x_0, x_1, \cdots) = 0$, where P is a polynomial with integer coefficients.

First we number the terms $\tau_0, \tau_1, \tau_2, \cdots$ built up from the variables and natural numbers by repeated additions and multiplications as follows:

$$\begin{aligned} \tau_{4n} &= n, \\ \tau_{4n+1} &= x_n, \\ \tau_{4n+2} &= \tau_{Kn} + \tau_{Ln}, \\ \tau_{4n+3} &= \tau_{Kn} \cdot \tau_{Ln}. \end{aligned}$$

Then we number the equations with the nth equation being $\tau_{Kn} = \tau_{Ln}$. We could also write the nth equation as $\tau_{Kn} - \tau_{Ln} = 0$. Thus, the eighth equation is $0 + 0 = x_0$ and the number of $2 \cdot x_0^2 = 1 + x_1^2$ is 7,697,614,550.

Finally, we wish to give a method of representing finite sequences of natural numbers due to Gödel. Let Rem (x, y) be the least nonnegative remainder of x divided by y.

LEMMA (GÖDEL 1931): *For every finite sequence $s_0, s_1, \cdots, s_k$ of natural numbers, there are natural numbers a and d such that*

(2.1) $\quad s_t = \text{Rem}(a, 1 + (t + 1)d)$ *for* $t = 0, \cdots, k$.

Proof: Now (2.1) is equivalent to

(2.2) $$a \equiv s_t \pmod{1 + (t + 1)d},$$
$$0 \leqq s_t < 1 + (t + 1)d \quad \text{for} \quad t = 0, \cdots, k.$$

By the Chinese remainder theorem, the congruences can be satisfied for some a if the moduli are relatively prime and the inequalities will be satisfied if d is sufficiently large. Take for d a sufficiently large multiple of $k!$ so that $d > s_t$ for all $t \leqq k$. If a prime p divides both $1 + (t+1)d$ and $1 + (t'+1)d$, then $p \mid (t - t')d$. If $t \neq t'$, then $0 < |t - t'| \leqq k$. Hence $p \mid k!$ and so $p \mid d$. But $p \nmid d$ since by hypothesis $p \mid 1 + (t+1)d$. Therefore for this choice of d, the moduli in (2.2) are relatively prime and a satisfying (2.2) exists.

This lemma is at the root of much of what follows.

3. COMPUTING AND LISTING

In Section 1 we spoke of a "general method" to tell if an arbitrary diophantine equation has a solution. What do we have in mind? We mean a set of instructions (necessarily finite) which describe in a completely deterministic way, how to start from an arbitrary diophantine equation $P(x_1, \cdots, x_k) = 0$ and to obtain after a finite number of steps the correct answer to the question: Does $P = 0$ have a solution? At no step in the process should the instructions call for either ingenuity or chance. On the other hand, we do not demand practicality of the method or place any restrictions on time or space needed to carry out the process.

Suppose we find a method to tell if an arbitrary diophantine equation has a solution. We give a proof that our method works, i.e., for every diophantine equation, it yields the correct answer to the question: Does this equation have a solution? Then if the argument is sound, everyone will presumably admit that Hilbert's problem is solved. We do not need to agree ahead of time on the exact meaning of the term "general method." But if there is no general method and we wish to prove that there is none, then we need to be precise. Hence until the notion of computability was defined, no one could ask if such a method exists.

Let us number all diophantine equations in some systematic way such as in the last section. Thus given any n, we can write down the nth equation and given an equation, we can figure out its

number (or at least one of its numbers if some equations occur more than once). Let $\mathcal{S}$ be the set of numbers of equations which have solutions. Then a method to tell whether or not an arbitrary natural number n belongs to $\mathcal{S}$ would also give us a method to tell whether or not an arbitrary diophantine equation has a solution.

We call a set of numbers *computable* if there is a method of deciding whether or not an arbitrary natural number n belongs to the set. A *method* consists of a finite set of instructions which for each natural number n specifies a calculation terminating in a "yes" or "no" answer to the question: Does n belong to the set being computed?

Clearly, this is not a mathematical definition of the class of computable sets but, rather, it is a description of the intuitive concept of a computable set. In Section 5 we will give a strictly mathematical characterization of the sets which are identified with this intuitive concept of computable set.

We call a set $\mathcal{L}$ of natural numbers *listable* if there is a method of making a list of the elements of $\mathcal{L}$. A *list* is either a finite or infinite sequence, with or without repetitions. For example, the set $\mathcal{S}$ of numbers of solvable diophantine equations is listable. To make the list, we simply try the possible values of the arguments in the equations in a systematic way. Each time we find a solution of some equation, we put its number on our list.

What is a method of listing a set of numbers? It is a set of instructions which provides for a completely deterministic calculation which may or may not terminate. From time to time during the calculation, a particular number is designated as the next number on the list and we place it next on the list.

We say a function $F(x_1, \cdots, x_k)$ defined for all natural numbers and assuming natural numbers as values is *computable* if there is a method to compute the value of $F(x_1, \cdots, x_k)$ for an arbitrary k-tuple of natural numbers. Here a method is a set of instructions such that if it is applied to any k-tuple of natural numbers $x_1, \cdots, x_k$ it will yield a completely deterministic calculation of the natural number which is $F(x_1, \cdots, x_k)$.

We will take the concept of *listable set* as our fundamental intuitive notion. The following observations show that the