ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*, it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers for explicit treatment as involving the least cumbrous technique. I hope shortly to give an account of the relations of the computable numbers, functions, and so forth to one another. This will include a development of the theory of functions of a real variable expressed in terms of computable numbers. According to my definition, a number is computable if its decimal can be written down by a machine.

In §§ 9, 10 I give some arguments with the intention of showing that the computable numbers include all numbers which could naturally be regarded as computable. In particular, I show that certain large classes of numbers are computable. They include, for instance, the real parts of all algebraic numbers, the real parts of the zeros of the Bessel functions, the numbers π, e, etc. The computable numbers do not, however, include all definable numbers, and an example is given of a definable number which is not computable.

Although the class of computable numbers is so great, and in many ways similar to the class of real numbers, it is nevertheless enumerable. In § 8 I examine certain arguments which would seem to prove the contrary. By the correct application of one of these arguments, conclusions are reached which are superficially similar to those of Gödel†. These results

† Gödel, "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I", *Monatshefte Math. Phys.*, 38 (1931), 173–198.

have valuable applications. In particular, it is shown (§ 11) that the Hilbertian Entscheidungsproblem can have no solution.

In a recent paper Alonzo Church† has introduced an idea of "effective calculability", which is equivalent to my "computability", but is very differently defined. Church also reaches similar conclusions about the Entscheidungsproblem‡. The proof of equivalence between "computability" and "effective calculability" is outlined in an appendix to the present paper.

1. *Computing machines.*

We have said that the computable numbers are those whose decimals are calculable by finite means. This requires rather more explicit definition. No real attempt will be made to justify the definitions given until we reach § 9. For the present I shall only say that the justification lies in the fact that the human memory is necessarily limited.

We may compare a man in the process of computing a real number to a machine which is only capable of a finite number of conditions $q_1, q_2, \ldots, q_R$ which will be called "m-configurations". The machine is supplied with a "tape" (the analogue of paper) running through it, and divided into sections (called "squares") each capable of bearing a "symbol". At any moment there is just one square, say the r-th, bearing the symbol $\mathfrak{S}(r)$ which is "in the machine". We may call this square the "scanned square". The symbol on the scanned square may be called the "scanned symbol". The "scanned symbol" is the only one of which the machine is, so to speak, "directly aware". However, by altering its m-configuration the machine can effectively remember some of the symbols which it has "seen" (scanned) previously. The possible behaviour of the machine at any moment is determined by the m-configuration q_n and the scanned symbol $\mathfrak{S}(r)$. This pair $q_n, \mathfrak{S}(r)$ will be called the "configuration": thus the configuration determines the possible behaviour of the machine. In some of the configurations in which the scanned square is blank (*i.e.* bears no symbol) the machine writes down a new symbol on the scanned square: in other configurations it erases the scanned symbol. The machine may also change the square which is being scanned, but only by shifting it one place to right or left. In addition to any of these operations the m-configuration may be changed. Some of the symbols written down

† Alonzo Church, "An unsolvable problem of elementary number theory", *American J. of Math.*, 58 (1936), 345–363.

‡ Alonzo Church, "A note on the Entscheidungsproblem", *J. of Symbolic Logic*, 1 (1936), 40–41.

will form the sequence of figures which is the decimal of the real number which is being computed. The others are just rough notes to "assist the memory". It will only be these rough notes which will be liable to erasure.

It is my contention that these operations include all those which are used in the computation of a number. The defence of this contention will be easier when the theory of the machines is familiar to the reader. In the next section I therefore proceed with the development of the theory and assume that it is understood what is meant by "machine", "tape", "scanned", etc.

2. *Definitions.*

Automatic machines.

If at each stage the motion of a machine (in the sense of § 1) is *completely* determined by the configuration, we shall call the machine an "automatic machine" (or *a*-machine).

For some purposes we might use machines (choice machines or *c*-machines) whose motion is only partially determined by the configuration (hence the use of the word "possible" in § 1). When such a machine reaches one of these ambiguous configurations, it cannot go on until some arbitrary choice has been made by an external operator. This would be the case if we were using machines to deal with axiomatic systems. In this paper I deal only with automatic machines, and will therefore often omit the prefix *a*-.

Computing machines.

If an *a*-machine prints two kinds of symbols, of which the first kind (called figures) consists entirely of 0 and 1 (the others being called symbols of the second kind), then the machine will be called a computing machine. If the machine is supplied with a blank tape and set in motion, starting from the correct initial *m*-configuration, the subsequence of the symbols printed by it which are of the first kind will be called the *sequence computed by the machine*. The real number whose expression as a binary decimal is obtained by prefacing this sequence by a decimal point is called the *number computed by the machine*.

At any stage of the motion of the machine, the number of the scanned square, the complete sequence of all symbols on the tape, and the *m*-configuration will be said to describe the *complete configuration* at that stage. The changes of the machine and tape between successive complete configurations will be called the *moves* of the machine.

Circular and circle-free machines.

If a computing machine never writes down more than a finite number of symbols of the first kind, it will be called *circular*. Otherwise it is said to be *circle-free*.

A machine will be circular if it reaches a configuration from which there is no possible move, or if it goes on moving, and possibly printing symbols of the second kind, but cannot print any more symbols of the first kind. The significance of the term "circular" will be explained in § 8.

Computable sequences and numbers.

A sequence is said to be computable if it can be computed by a circle-free machine. A number is computable if it differs by an integer from the number computed by a circle-free machine.

We shall avoid confusion by speaking more often of computable sequences than of computable numbers.

3. *Examples of computing machines.*

I. A machine can be constructed to compute the sequence 010101 .... The machine is to have the four m-configurations "$\mathfrak{b}$", "$\mathfrak{c}$", "$\mathfrak{f}$", "$\mathfrak{e}$" and is capable of printing "0" and "1". The behaviour of the machine is described in the following table in which "R" means "the machine moves so that it scans the square immediately on the right of the one it was scanning previously". Similarly for "L". "E" means "the scanned symbol is erased" and "P" stands for "prints". This table (and all succeeding tables of the same kind) is to be understood to mean that for a configuration described in the first two columns the operations in the third column are carried out successively, and the machine then goes over into the m-configuration described in the last column. When the second column is left blank, it is understood that the behaviour of the third and fourth columns applies for any symbol and for no symbol. The machine starts in the m-configuration $\mathfrak{b}$ with a blank tape.

Configuration		*Behaviour*	
m-config.	*symbol*	*operations*	*final m-config.*
$\mathfrak{b}$	None	$P0$, R	$\mathfrak{c}$
$\mathfrak{c}$	None	R	$\mathfrak{c}$
$\mathfrak{e}$	None	$P1$, R	$\mathfrak{f}$
$\mathfrak{f}$	None	R	$\mathfrak{b}$

If (contrary to the description in § 1) we allow the letters L, R to appear more than once in the operations column we can simplify the table considerably.

m-config.	*symbol*	*operations*	*final m-config.*
𝔟	None	$P0$	𝔟
	0	$R, R, P1$	𝔟
	1	$R, R, P0$	𝔟

II. As a slightly more difficult example we can construct a machine to compute the sequence 00101101111011111011111 .... The machine is to be capable of five m-configurations, viz. "𝔬", "𝔮", "𝔭", "𝔣", "𝔟" and of printing "ə", "x", "0", "1". The first three symbols on the tape will be "ə ə 0"; the other figures follow on alternate squares. On the intermediate squares we never print anything but "x". These letters serve to "keep the place" for us and are erased when we have finished with them. We also arrange that in the sequence of figures on alternate squares there shall be no blanks.

Configuration		*Behaviour*	
m-config.	*symbol*	*operations*	*final m-config.*
𝔟		Pə, R, Pə, R, $P0$, R, R, $P0$, L, L	𝔬
𝔬	1	R, Px, L, L, L	𝔬
	0		𝔮
𝔮	Any (0 or 1)	R, R	𝔮
	None	$P1$, L	𝔭
𝔭	x	E, R	𝔮
	ə	R	𝔣
	None	L, L	𝔭
𝔣	Any	R, R	𝔣
	None	$P0, L, L$	𝔬

To illustrate the working of this machine a table is given below of the first few complete configurations. These complete configurations are described by writing down the sequence of symbols which are on the tape,

with the m-configuration written below the scanned symbol. The successive complete configurations are separated by colons.

: ə ə 0 0 : ə ə 0 0 : ə ə 0 0 : ə ə 0 0 : ə ə 0 0 1 :
ɓ ɔ ɳ ɳ ɳ ƥ
ə ə 0 0 1 : ə ə 0 0 1 : ə ə 0 0 1 : ə ə 0 0 1 :
ƥ ƥ f f
ə ə 0 0 1 : ə ə 0 0 1 : ə ə 0 0 1 0 :
f f ɔ
ə ə 0 0 1 x 0 : ....
ɔ

This table could also be written in the form

ɓ : ə ə ɔ 0 0 : ə ə ɳ 0 0 : .... (C)

in which a space has been made on the left of the scanned symbol and the m-configuration written in this space. This form is less easy to follow, but we shall make use of it later for theoretical purposes.

The convention of writing the figures only on alternate squares is very useful: I shall always make use of it. I shall call the one sequence of alternate squares F-squares and the other sequence E-squares. The symbols on E-squares will be liable to erasure. The symbols on F-squares form a continuous sequence. There are no blanks until the end is reached. There is no need to have more than one E-square between each pair of F-squares: an apparent need of more E-squares can be satisfied by having a sufficiently rich variety of symbols capable of being printed on E-squares. If a symbol β is on an F-square S and a symbol α is on the E-square next on the right of S, then S and β will be said to be *marked* with α. The process of printing this α will be called marking β (or S) with α.

4. *Abbreviated tables.*

There are certain types of process used by nearly all machines, and these, in some machines, are used in many connections. These processes include copying down sequences of symbols, comparing sequences, erasing all symbols of a given form, etc. Where such processes are concerned we can abbreviate the tables for the m-configurations considerably by the use of "skeleton tables". In skeleton tables there appear capital German letters and small Greek letters. These are of the nature of "variables". By replacing each capital German letter throughout by an m-configuration

and each small Greek letter by a symbol, we obtain the table for an m-configuration.

The skeleton tables are to be regarded as nothing but abbreviations: they are not essential. So long as the reader understands how to obtain the complete tables from the skeleton tables, there is no need to give any exact definitions in this connection.

Let us consider an example:

m-config.	*Symbol*	*Behaviour*		*Final m-config.*
$\mathfrak{f}(\mathfrak{C}, \mathfrak{B}, \alpha)$	$\mathfrak{d}$	L		$\mathfrak{f}_1(\mathfrak{C}, \mathfrak{B}, \alpha)$
	not $\mathfrak{d}$	L		$\mathfrak{f}(\mathfrak{C}, \mathfrak{B}, \alpha)$
$\mathfrak{f}_1(\mathfrak{C}, \mathfrak{B}, \alpha)$	α			$\mathfrak{C}$
	not α	R		$\mathfrak{f}_1(\mathfrak{C}, \mathfrak{B}, \alpha)$
	None	R		$\mathfrak{f}_2(\mathfrak{C}, \mathfrak{B}, \alpha)$
$\mathfrak{f}_2(\mathfrak{C}, \mathfrak{B}, \alpha)$	α			$\mathfrak{C}$
	not α	R		$\mathfrak{f}_1(\mathfrak{C}, \mathfrak{B}, \alpha)$
	None	R		$\mathfrak{B}$

From the m-configuration $\mathfrak{f}(\mathfrak{C}, \mathfrak{B}, \alpha)$ the machine finds the symbol of form α which is farthest to the left (the "first α") and the m-configuration then becomes $\mathfrak{C}$. If there is no α then the m-configuration becomes $\mathfrak{B}$.

If we were to replace $\mathfrak{C}$ throughout by $\mathfrak{q}$ (say), $\mathfrak{B}$ by $\mathfrak{r}$, and α by x, we should have a complete table for the m-configuration $\mathfrak{f}(\mathfrak{q}, \mathfrak{r}, x)$. $\mathfrak{f}$ is called an "m-configuration function" or "m-function".

The only expressions which are admissible for substitution in an m-function are the m-configurations and symbols of the machine. These have to be enumerated more or less explicitly: they may include expressions such as $\mathfrak{p}(\mathfrak{c}, x)$; indeed they must if there are any m-functions used at all. If we did not insist on this explicit enumeration, but simply stated that the machine had certain m-configurations (enumerated) and all m-configurations obtainable by substitution of m-configurations in certain m-functions, we should usually get an infinity of m-configurations; *e.g.*, we might say that the machine was to have the m-configuration $\mathfrak{q}$ and all m-configurations obtainable by substituting an m-configuration for $\mathfrak{C}$ in $\mathfrak{p}(\mathfrak{C})$. Then it would have $\mathfrak{q}$, $\mathfrak{p}(\mathfrak{q})$, $\mathfrak{p}\left(\mathfrak{p}(\mathfrak{q})\right)$, $\mathfrak{p}\left(\mathfrak{p}\left(\mathfrak{p}(\mathfrak{q})\right)\right)$, ... as m-configurations.

Our interpretation rule then is this. We are given the names of the m-configurations of the machine, mostly expressed in terms of m-functions. We are also given skeleton tables. All we want is the complete table for the m-configurations of the machine. This is obtained by repeated substitution in the skeleton tables.

Further examples.

(In the explanations the symbol "→" is used to signify "the machine goes into the m-configuration. . . .")

$\mathfrak{e}(\mathfrak{C}, \mathfrak{B}, \mathfrak{a})$		$\mathfrak{f}\left(\mathfrak{e}_1(\mathfrak{C}, \mathfrak{B}, \mathfrak{a}), \mathfrak{B}, \mathfrak{a}\right)$	From $\mathfrak{e}(\mathfrak{C}, \mathfrak{B}, \mathfrak{a})$ the first $\mathfrak{a}$ is erased and $\rightarrow \mathfrak{C}$. If there is no $\mathfrak{a} \rightarrow \mathfrak{B}$.
$\mathfrak{e}_1(\mathfrak{C}, \mathfrak{B}, \mathfrak{a})$	E	$\mathfrak{C}$	
$\mathfrak{e}(\mathfrak{B}, \mathfrak{a})$		$\mathfrak{e}\left(\mathfrak{e}(\mathfrak{B}, \mathfrak{a}), \mathfrak{B}, \mathfrak{a}\right)$	From $\mathfrak{e}(\mathfrak{B}, \mathfrak{a})$ all letters $\mathfrak{a}$ are erased and $\rightarrow \mathfrak{B}$.

The last example seems somewhat more difficult to interpret than most. Let us suppose that in the list of m-configurations of some machine there appears $\mathfrak{e}(\mathfrak{b}, x)$ $(= \mathfrak{q}$, say). The table is

	$\mathfrak{e}(\mathfrak{b}, x)$	$\mathfrak{e}\left(\mathfrak{e}(\mathfrak{b}, x), \mathfrak{b}, x\right)$
or	$\mathfrak{q}$	$\mathfrak{e}(\mathfrak{q}, \mathfrak{b}, x)$.

Or, in greater detail:

	$\mathfrak{q}$	$\mathfrak{e}(\mathfrak{q}, \mathfrak{b}, x)$
	$\mathfrak{e}(\mathfrak{q}, \mathfrak{b}, x)$	$\mathfrak{f}\left(\mathfrak{e}_1(\mathfrak{q}, \mathfrak{b}, x), \mathfrak{b}, x\right)$
	$\mathfrak{e}_1(\mathfrak{q}, \mathfrak{b}, x)$ E	$\mathfrak{q}$.

In this we could replace $\mathfrak{e}_1(\mathfrak{q}, \mathfrak{b}, x)$ by $\mathfrak{q}'$ and then give the table for $\mathfrak{f}$ (with the right substitutions) and eventually reach a table in which no m-functions appeared.

$\mathfrak{pe}(\mathfrak{C}, \beta)$			$\mathfrak{f}\left(\mathfrak{pe}_1(\mathfrak{C}, \beta), \mathfrak{C}, \mathfrak{a}\right)$	From $\mathfrak{pe}(\mathfrak{C}, \beta)$ the machine prints β at the end of the sequence of symbols and $\rightarrow \mathfrak{C}$.
$\mathfrak{pe}_1(\mathfrak{C}, \beta)$	Any	R, R	$\mathfrak{pe}_1(\mathfrak{C}, \beta)$	
	None	$P\beta$	$\mathfrak{C}$	
$\mathfrak{l}(\mathfrak{C})$		L	$\mathfrak{C}$	From $\mathfrak{f}'(\mathfrak{C}, \mathfrak{B}, \mathfrak{a})$ it does the same as for $\mathfrak{f}(\mathfrak{C}, \mathfrak{B}, \mathfrak{a})$ but moves to the left before $\rightarrow \mathfrak{C}$.
$\mathfrak{r}(\mathfrak{C})$		R	$\mathfrak{C}$	
$\mathfrak{f}'(\mathfrak{C}, \mathfrak{B}, \mathfrak{a})$			$\mathfrak{f}\left(\mathfrak{l}(\mathfrak{C}), \mathfrak{B}, \mathfrak{a}\right)$	
$\mathfrak{f}''(\mathfrak{C}, \mathfrak{B}, \mathfrak{a})$			$\mathfrak{f}\left(\mathfrak{r}(\mathfrak{C}), \mathfrak{B}, \mathfrak{a}\right)$	
$\mathfrak{c}(\mathfrak{C}, \mathfrak{B}, \mathfrak{a})$			$\mathfrak{f}'\left(\mathfrak{c}_1(\mathfrak{C}), \mathfrak{B}, \mathfrak{a}\right)$	$\mathfrak{c}(\mathfrak{C}, \mathfrak{B}, \mathfrak{a})$. The machine writes at the end the first symbol marked $\mathfrak{a}$ and $\rightarrow \mathfrak{C}$.
$\mathfrak{c}_1(\mathfrak{C})$	β		$\mathfrak{pe}(\mathfrak{C}, \beta)$	

The last line stands for the totality of lines obtainable from it by replacing β by any symbol which may occur on the tape of the machine concerned.

$\mathfrak{ce}(\mathfrak{C}, \mathfrak{B}, \alpha)$ $\quad \mathfrak{c}\left(\mathfrak{e}(\mathfrak{C}, \mathfrak{B}, \alpha), \mathfrak{B}, \alpha\right)$

$\mathfrak{ce}(\mathfrak{B}, \alpha)$ $\quad \mathfrak{ce}\left(\mathfrak{ce}(\mathfrak{B}, \alpha), \mathfrak{B}, \alpha\right)$

$\mathfrak{ce}(\mathfrak{B}, \alpha)$. The machine copies down in order at the end all symbols marked α and erases the letters α; $\rightarrow \mathfrak{B}$.

$\mathfrak{re}(\mathfrak{C}, \mathfrak{B}, \alpha, \beta)$ $\quad \mathfrak{f}\left(\mathfrak{re}_1(\mathfrak{C}, \mathfrak{B}, \alpha, \beta), \mathfrak{B}, \alpha\right)$

$\mathfrak{re}_1(\mathfrak{C}, \mathfrak{B}, \alpha, \beta)$ $\quad E, P\beta$ $\quad \mathfrak{C}$

$\mathfrak{re}(\mathfrak{B}, \alpha, \beta)$ $\quad \mathfrak{re}\left(\mathfrak{re}(\mathfrak{B}, \alpha, \beta), \mathfrak{B}, \alpha, \beta\right)$

$\mathfrak{re}(\mathfrak{C}, \mathfrak{B}, \alpha, \beta)$. The machine replaces the first α by β and $\rightarrow \mathfrak{C} \rightarrow \mathfrak{B}$ if there is no α.

$\mathfrak{re}(\mathfrak{B}, \alpha, \beta)$. The machine replaces all letters α by β; $\rightarrow \mathfrak{B}$.

$\mathfrak{cr}(\mathfrak{C}, \mathfrak{B}, \alpha)$ $\quad \mathfrak{c}\left(\mathfrak{re}(\mathfrak{C}, \mathfrak{B}, \alpha, a), \mathfrak{B}, \alpha\right)$

$\mathfrak{cr}(\mathfrak{B}, \alpha)$ $\quad \mathfrak{cr}\left(\mathfrak{cr}(\mathfrak{B}, \alpha), \mathfrak{re}(\mathfrak{B}, a, \alpha), \alpha\right)$

$\mathfrak{cr}(\mathfrak{B}, \alpha)$ differs from $\mathfrak{ce}(\mathfrak{B}, \alpha)$ only in that the letters α are not erased. The m-configuration $\mathfrak{cr}(\mathfrak{B}, \alpha)$ is taken up when no letters "α" are on the tape.

$\mathfrak{cp}(\mathfrak{C}, \mathfrak{A}, \mathfrak{E}, \alpha, \beta)$ $\quad \mathfrak{f}'\left(\mathfrak{cp}_1(\mathfrak{C}_1 \mathfrak{A}, \beta), \mathfrak{f}(\mathfrak{A}, \mathfrak{E}, \beta), \alpha\right)$

$\mathfrak{cp}_1(\mathfrak{C}, \mathfrak{A}, \beta)$ $\quad \gamma$ $\quad \mathfrak{f}'\left(\mathfrak{cp}_2(\mathfrak{C}, \mathfrak{A}, \gamma), \mathfrak{A}, \beta\right)$

$\mathfrak{cp}_2(\mathfrak{C}, \mathfrak{A}, \gamma)$ $\quad \begin{cases} \gamma & \mathfrak{C} \\ \text{not } \gamma & \mathfrak{A}. \end{cases}$

The first symbol marked α and the first marked β are compared. If there is neither α nor β, $\rightarrow \mathfrak{E}$. If there are both and the symbols are alike, $\rightarrow \mathfrak{C}$. Otherwise $\rightarrow \mathfrak{A}$.

$\mathfrak{cpe}(\mathfrak{C}, \mathfrak{A}, \mathfrak{E}, \alpha, \beta)$ $\quad \mathfrak{cp}\left(\mathfrak{e}\left(\mathfrak{e}(\mathfrak{C}, \mathfrak{C}, \beta), \mathfrak{C}, \alpha\right), \mathfrak{A}, \mathfrak{E}, \alpha, \beta\right)$

$\mathfrak{cpe}(\mathfrak{C}, \mathfrak{A}, \mathfrak{E}, \alpha, \beta)$ differs from $\mathfrak{cp}(\mathfrak{C}, \mathfrak{A}, \mathfrak{E}, \alpha, \beta)$ in that in the case when there is similarity the first α and β are erased.

$\mathfrak{cpe}(\mathfrak{A}, \mathfrak{E}, \alpha, \beta)$ $\quad \mathfrak{cpe}\left(\mathfrak{cpe}(\mathfrak{A}, \mathfrak{E}, \alpha, \beta), \mathfrak{A}, \mathfrak{E}, \alpha, \beta\right)$.

$\mathfrak{cpe}(\mathfrak{A}, \mathfrak{E}, \alpha, \beta)$. The sequence of symbols marked α is compared with the sequence marked β. $\rightarrow \mathfrak{E}$ if they are similar. Otherwise $\rightarrow \mathfrak{A}$. Some of the symbols α and β are erased.

$\mathfrak{q}(\mathfrak{C})$	Any	R	$\mathfrak{q}(\mathfrak{C})$	$\mathfrak{q}(\mathfrak{C}, \alpha)$. The machine finds the last symbol of form α. $\rightarrow \mathfrak{C}$.
	None	R	$\mathfrak{q}_1(\mathfrak{C})$	
$\mathfrak{q}_1(\mathfrak{C})$	Any	R	$\mathfrak{q}(\mathfrak{C})$	
	None		$\mathfrak{C}$	
$\mathfrak{q}(\mathfrak{C}, \alpha)$			$\mathfrak{q}\left(\mathfrak{q}_1(\mathfrak{C}, \alpha)\right)$	
$\mathfrak{q}_1(\mathfrak{C}, \alpha)$	α		$\mathfrak{C}$	
	not α	L	$\mathfrak{q}_1(\mathfrak{C}, \alpha)$	
$\mathfrak{pe}_2(\mathfrak{C}, \alpha, \beta)$			$\mathfrak{pe}\left(\mathfrak{pe}(\mathfrak{C}, \beta), \alpha\right)$	$\mathfrak{pe}_2(\mathfrak{C}, \alpha, \beta)$. The machine prints α β at the end.
$\mathfrak{ce}_2(\mathfrak{B}, \alpha, \beta)$			$\mathfrak{ce}\left(\mathfrak{ce}(\mathfrak{B}, \beta), \alpha\right)$	$\mathfrak{ce}_3(\mathfrak{B}, \alpha, \beta, \gamma)$. The machine copies down at the end first the symbols marked α, then those marked β, and finally those marked γ; it erases the symbols α, β, γ.
$\mathfrak{ce}_3(\mathfrak{B}, \alpha, \beta, \gamma)$			$\mathfrak{ce}\left(\mathfrak{ce}_2(\mathfrak{B}, \beta, \gamma), \alpha\right)$	
$\mathfrak{e}(\mathfrak{C})$	$\mathfrak{d}$	R	$\mathfrak{e}_1(\mathfrak{C})$	From $\mathfrak{e}(\mathfrak{C})$ the marks are erased from all marked symbols. $\rightarrow \mathfrak{C}$.
	Not $\mathfrak{d}$	L	$\mathfrak{e}(\mathfrak{C})$	
$\mathfrak{e}_1(\mathfrak{C})$	Any	R, E, R	$\mathfrak{e}_1(\mathfrak{C})$	
	None		$\mathfrak{C}$	

5. *Enumeration of computable sequences.*

A computable sequence γ is determined by a description of a machine which computes γ. Thus the sequence 001011011101111... is determined by the table on p. 234, and, in fact, any computable sequence is capable of being described in terms of such a table.

It will be useful to put these tables into a kind of standard form. In the first place let us suppose that the table is given in the same form as the first table, for example, I on p. 233. That is to say, that the entry in the operations column is always of one of the forms $E : E$, $R : E$, $L : Pa : Pa$, $R : Pa$, $L : R : L :$ or no entry at all. The table can always be put into this form by introducing more m-configurations. Now let us give numbers to the m-configurations, calling them q_1, ..., q_R, as in § 1. The initial m-configuration is always to be called q_1. We also give numbers to the symbols S_1, ..., S_m

and, in particular, blank $= S_0$, $0 = S_1$, $1 = S_2$. The lines of the table are now of form

m-config.	*Symbol*	*Operations*	*Final m-config.*	
q_i	S_j	PS_k, L	q_m	(N_1)
q_i	S_j	PS_k, R	q_m	(N_2)
q_i	S_j	PS_k	q_m	(N_3)

Lines such as

q_i	S_j	E, R	q_m

are to be written as

q_i	S_j	PS_0, R	q_m

and lines such as

q_i	S_j	R	q_m

to be written as

q_i	S_j	PS_j, R	q_m

In this way we reduce each line of the table to a line of one of the forms (N_1), (N_2), (N_3).

From each line of form (N_1) let us form an expression $q_i S_j S_k L q_m$; from each line of form (N_2) we form an expression $q_i S_j S_k R q_m$; and from each line of form (N_3) we form an expression $q_i S_j S_k N q_m$.

Let us write down all expressions so formed from the table for the machine and separate them by semi-colons. In this way we obtain a complete description of the machine. In this description we shall replace q_i by the letter "D" followed by the letter "A" repeated i times, and S_j by "D" followed by "C" repeated j times. This new description of the machine may be called the *standard description* (S.D). It is made up entirely from the letters "A", "C", "D", "L", "R", "N", and from "$;$".

If finally we replace "A" by "1", "C" by "2", "D" by "3", "L" by "4", "R" by "5", "N" by "6", and "$;$" by "7" we shall have a description of the machine in the form of an arabic numeral. The integer represented by this numeral may be called a *description number* (D.N) of the machine. The D.N determine the S.D and the structure of the

machine uniquely. The machine whose D.N is n may be described as $\mathcal{M}(n)$.

To each computable sequence there corresponds at least one description number, while to no description number does there correspond more than one computable sequence. The computable sequences and numbers are therefore enumerable.

Let us find a description number for the machine I of § 3. When we rename the m-configurations its table becomes:

q_1	S_0	PS_1, R	q_2
q_2	S_0	PS_0, R	q_3
q_3	S_0	PS_2, R	q_4
q_4	S_0	PS_0, R	q_1

Other tables could be obtained by adding irrelevant lines such as

q_1	S_1	PS_1, R	q_2

Our first standard form would be

$$q_1 S_0 S_1 R q_2;\ q_2 S_0 S_0 R q_3;\ q_3 S_0 S_2 R q_4;\ q_4 S_0 S_0 R q_1;.$$

The standard description is

$$DADDCRDAA;DAADDRDAAA;$$

$$DAAADDCCRDAAAA;DAAAADDRDA;$$

A description number is

3133253117311335311173111332253111173111133531732325311 7

and so is

313325311731133531117311133225311117311113353173132325311 7

A number which is a description number of a circle-free machine will be called a *satisfactory* number. In § 8 it is shown that there can be no general process for determining whether a given number is satisfactory or not.

6. *The universal computing machine.*

It is possible to invent a single machine which can be used to compute any computable sequence. If this machine $\mathcal{U}$ is supplied with a tape on the beginning of which is written the S.D of some computing machine $\mathcal{M}$,

then $\mathfrak{U}$ will compute the same sequence as $\mathcal{M}$. In this section I explain in outline the behaviour of the machine. The next section is devoted to giving the complete table for $\mathfrak{U}$.

Let us first suppose that we have a machine $\mathcal{M}'$ which will write down on the F-squares the successive complete configurations of $\mathcal{M}$. These might be expressed in the same form as on p. 235, using the second description, (C), with all symbols on one line. Or, better, we could transform this description (as in § 5) by replacing each m-configuration by "D" followed by "A" repeated the appropriate number of times, and by replacing each symbol by "D" followed by "C" repeated the appropriate number of times. The numbers of letters "A" and "C" are to agree with the numbers chosen in § 5, so that, in particular, "0" is replaced by "DC", "1" by "DCC", and the blanks by "D". These substitutions are to be made after the complete configurations have been put together, as in (C). Difficulties arise if we do the substitution first. In each complete configuration the blanks would all have to be replaced by "D", so that the complete configuration would not be expressed as a finite sequence of symbols.

If in the description of the machine II of § 3 we replace "$\mathfrak{o}$" by "DAA", "$\mathfrak{e}$" by "$DCCC$", "$\mathfrak{q}$" by "$DAAA$", then the sequence (C) becomes:

$$DA:DCCCDCCCDAADCDDC:DCCCDCCCDAAADCDDC:\ldots \quad (C_1)$$

(This is the sequence of symbols on F-squares.)

It is not difficult to see that if $\mathcal{M}$ can be constructed, then so can $\mathcal{M}'$. The manner of operation of $\mathcal{M}'$ could be made to depend on having the rules of operation (*i.e.*, the S.D) of $\mathcal{M}$ written somewhere within itself (*i.e.* within $\mathcal{M}'$); each step could be carried out by referring to these rules. We have only to regard the rules as being capable of being taken out and exchanged for others and we have something very akin to the universal machine.

One thing is lacking: at present the machine $\mathcal{M}'$ prints no figures. We may correct this by printing between each successive pair of complete configurations the figures which appear in the new configuration but not in the old. Then (C_1) becomes

$$DDA:0:0:DCCCDCCCDAADCDDC:DCCC\ldots. \quad (C_2)$$

It is not altogether obvious that the E-squares leave enough room for the necessary "rough work", but this is, in fact, the case.

The sequences of letters between the colons in expressions such as (C_1) may be used as standard descriptions of the complete configurations. When the letters are replaced by figures, as in § 5, we shall have a numerical

description of the complete configuration, which may be called its description number.

7. *Detailed description of the universal machine.*

A table is given below of the behaviour of this universal machine. The m-configurations of which the machine is capable are all those occurring in the first and last columns of the table, together with all those which occur when we write out the unabbreviated tables of those which appear in the table in the form of m-functions. *E.g.*, $\mathfrak{e}(\mathfrak{anf})$ appears in the table and is an m-function. Its unabbreviated table is (see p. 239)

$\mathfrak{e}(\mathfrak{anf})$	ə	R	$\mathfrak{e}_1(\mathfrak{anf})$
	not ə	L	$\mathfrak{e}(\mathfrak{anf})$
$\mathfrak{e}_1(\mathfrak{anf})$	Any	R, E, R	$\mathfrak{e}_1(\mathfrak{anf})$
	None		$\mathfrak{anf}$

Consequently $\mathfrak{e}_1(\mathfrak{anf})$ is an m-configuration of $\mathfrak{U}$.

When $\mathfrak{U}$ is ready to start work the tape running through it bears on it the symbol ə on an F-square and again ə on the next E-square; after this, on F-squares only, comes the S.D of the machine followed by a double colon "::" (a single symbol, on an F-square). The S.D consists of a number of instructions, separated by semi-colons.

Each instruction consists of five consecutive parts

(i) "D" followed by a sequence of letters "A". This describes the relevant m-configuration.

(ii) "D" followed by a sequence of letters "C". This describes the scanned symbol.

(iii) "D" followed by another sequence of letters "C". This describes the symbol into which the scanned symbol is to be changed.

(iv) "L", "R", or "N", describing whether the machine is to move to left, right, or not at all.

(v) "D" followed by a sequence of letters "A". This describes the final m-configuration.

The machine $\mathfrak{U}$ is to be capable of printing "A", "C", "D", "0", "1", "u", "v", "w", "x", "y", "z". The S.D is formed from ";", "A", "C", "D", "L", "R", "N".

Subsidiary skeleton table.

$\mathrm{con}(\mathfrak{C}, \alpha)$	Not A	R, R	$\mathrm{con}(\mathfrak{C}, \alpha)$
	A	$L, P\alpha, R$	$\mathrm{con}_1(\mathfrak{C}, \alpha)$
$\mathrm{con}_1(\mathfrak{C}, \alpha)$	A	$R, P\alpha, R$	$\mathrm{con}_1(\mathfrak{C}, \alpha)$
	D	$R, P\alpha, R$	$\mathrm{con}_2(\mathfrak{C}, \alpha)$
$\mathrm{con}_2(\mathfrak{C}, \alpha)$	C	$R, P\alpha, R$	$\mathrm{con}_2(\mathfrak{C}, \alpha)$
	Not C	R, R	$\mathfrak{C}$

$\mathrm{con}(\mathfrak{C}, \alpha)$. Starting from an F-square, S say, the sequence C of symbols describing a configuration closest on the right of S is marked out with letters α. $\rightarrow \mathfrak{C}$.

$\mathrm{con}(\mathfrak{C},)$. In the final configuration the machine is scanning the square which is four squares to the right of the last square of C. C is left unmarked.

The table for $\mathfrak{U}$.

$\mathfrak{b}$			$\mathfrak{f}(\mathfrak{b}_1, \mathfrak{b}_1, ::)$
$\mathfrak{b}_1$	$R, R, P:, R, R, PD, R, R, PA$		$\mathfrak{anf}$
$\mathfrak{anf}$			$\mathfrak{g}(\mathfrak{anf}_1, :)$
$\mathfrak{anf}_1$			$\mathrm{con}(\mathfrak{fom}, y)$
$\mathfrak{fom}$	;	R, Pz, L	$\mathrm{con}(\mathfrak{fmp}, x)$
	z	L, L	$\mathfrak{fom}$
	not z nor ;	L	$\mathfrak{fom}$
$\mathfrak{fmp}$			$\mathrm{cpe}\big(\mathfrak{c}(\mathfrak{fom}, x, y), \mathfrak{sim}, x, y\big)$

$\mathfrak{b}$. The machine prints $:DA$ on the F-squares after $:: \rightarrow \mathfrak{anf}$.

$\mathfrak{anf}$. The machine marks the configuration in the last complete configuration with y. $\rightarrow \mathfrak{fom}$.

$\mathfrak{fom}$. The machine finds the last semi-colon not marked with z. It marks this semi-colon with z and the configuration following it with x.

$\mathfrak{fmp}$. The machine compares the sequences marked x and y. It erases all letters x and y. $\rightarrow \mathfrak{sim}$ if they are alike. Otherwise $\rightarrow \mathfrak{fom}$.

$\mathfrak{anf}$. Taking the long view, the last instruction relevant to the last configuration is found. It can be recognised afterwards as the instruction following the last semi-colon marked z. $\rightarrow \mathfrak{sim}$.

$\mathfrak{sim}$ $\quad$ $\mathfrak{f}'(\mathfrak{sim}_1, \mathfrak{sim}_1, z)$

$\mathfrak{sim}_1$ $\quad$ $\mathrm{con}(\mathfrak{sim}_2,)$

$\mathfrak{sim}_2$

A		$\mathfrak{sim}_3$
not A	R, Pu, R, R, R	$\mathfrak{sim}_2$

$\mathfrak{sim}_3$

not A	L, Py	$\mathfrak{e}(\mathfrak{mf}, z)$
A	L, Py, R, R, R	$\mathfrak{sim}_3$

$\mathfrak{sim}$. The machine marks out the instructions. That part of the instructions which refers to operations to be carried out is marked with u, and the final m-configuration with y. The letters z are erased.

$\mathfrak{mf}$ $\quad$ $\mathfrak{g}(\mathfrak{mf}, :)$

$\mathfrak{mf}_1$

not A	R, R	$\mathfrak{mf}_1$
A	L, L, L, L	$\mathfrak{mf}_2$

$\mathfrak{mf}_2$

C	R, Px, L, L, L	$\mathfrak{mf}_2$
:		$\mathfrak{mf}_4$
D	R, Px, L, L, L	$\mathfrak{mf}_3$

$\mathfrak{mf}_3$

not :	R, Pv, L, L, L	$\mathfrak{mf}_3$
:		$\mathfrak{mf}_4$

$\mathfrak{mf}$. The last complete configuration is marked out into four sections. The configuration is left unmarked. The symbol directly preceding it is marked with x. The remainder of the complete configuration is divided into two parts, of which the first is marked with v and the last with w. A colon is printed after the whole. $\rightarrow \mathfrak{sh}$.

$\mathfrak{mf}_4$ $\quad$ $\mathrm{con}\left(\mathfrak{l}\left(\mathfrak{l}(\mathfrak{mf}_5)\right), \right)$

$\mathfrak{mf}_5$

Any	R, Pw, R	$\mathfrak{mf}_5$
None	$P:$	$\mathfrak{sh}$

$\mathfrak{sh}$ $\quad$ $\mathfrak{f}(\mathfrak{sh}_1, \mathfrak{inst}, u)$

$\mathfrak{sh}_1$ $\quad$ L, L, L $\quad$ $\mathfrak{sh}_2$

$\mathfrak{sh}_2$

D	R, R, R, R	$\mathfrak{sh}_2$
not D		$\mathfrak{inst}$

$\mathfrak{sh}_3$

C	R, R	$\mathfrak{sh}_4$
not C		$\mathfrak{inst}$

$\mathfrak{sh}_4$

C	R, R	$\mathfrak{sh}_5$
not C		$\mathfrak{pe}_2(\mathfrak{inst}, 0, :)$

$\mathfrak{sh}_5$

C		$\mathfrak{inst}$
not C		$\mathfrak{pe}_2(\mathfrak{inst}, 1, :)$

$\mathfrak{sh}$. The instructions (marked u) are examined. If it is found that they involve "Print 0" or "Print 1", then 0: or 1: is printed at the end.

inst			$g\left(\mathfrak{l}(\mathfrak{inst}_1), u\right)$
$\mathfrak{inst}_1$	a	R, E	$\mathfrak{inst}_1(a)$
$\mathfrak{inst}_1(L)$			$\mathfrak{ce}_5(\mathfrak{ov}, v, y, x, u, w)$
$\mathfrak{inst}_1(R)$			$\mathfrak{ce}_5(\mathfrak{ov}, v, x, u, y, w)$
$\mathfrak{inst}_1(N)$			$\mathfrak{ce}_5(\mathfrak{ov}, v, x, y, u, w)$
$\mathfrak{ov}$			$\mathfrak{e}(\mathfrak{anf})$

inst. The next complete configuration is written down, carrying out the marked instructions. The letters u, v, w, x, y are erased. $\rightarrow$anf.

8. *Application of the diagonal process.*

It may be thought that arguments which prove that the real numbers are not enumerable would also prove that the computable numbers and sequences cannot be enumerable*. It might, for instance, be thought that the limit of a sequence of computable numbers must be computable. This is clearly only true if the sequence of computable numbers is defined by some rule.

Or we might apply the diagonal process. "If the computable sequences are enumerable, let α_n be the n-th computable sequence, and let $\phi_n(m)$ be the m-th figure in α_n. Let β be the sequence with $1-\phi_n(n)$ as its n-th figure. Since β is computable, there exists a number K such that $1-\phi_n(n) = \phi_K(n)$ all n. Putting $n = K$, we have $1 = 2\phi_K(K)$, *i.e.* 1 is even. This is impossible. The computable sequences are therefore not enumerable".

The fallacy in this argument lies in the assumption that β is computable. It would be true if we could enumerate the computable sequences by finite means, but the problem of enumerating computable sequences is equivalent to the problem of finding out whether a given number is the D.N of a circle-free machine, and we have no general process for doing this in a finite number of steps. In fact, by applying the diagonal process argument correctly, we can show that there cannot be any such general process.

The simplest and most direct proof of this is by showing that, if this general process exists, then there is a machine which computes β. This proof, although perfectly sound, has the disadvantage that it may leave the reader with a feeling that "there must be something wrong". The proof which I shall give has not this disadvantage, and gives a certain insight into the significance of the idea "circle-free". It depends not on constructing β, but on constructing β', whose n-th figure is $\phi_n(n)$.

* *Cf.* Hobson, *Theory of functions of a real variable* (2nd ed., 1921), 87, 88.

Let us suppose that there is such a process; that is to say, that we can invent a machine $\mathfrak{D}$ which, when supplied with the S.D of any computing machine $\mathcal{M}$ will test this S.D and if $\mathcal{M}$ is circular will mark the S.D with the symbol "u" and if it is circle-free will mark it with "s". By combining the machines $\mathfrak{D}$ and $\mathfrak{U}$ we could construct a machine $\mathcal{H}$ to compute the sequence β'. The machine $\mathfrak{D}$ may require a tape. We may suppose that it uses the E-squares beyond all symbols on F-squares, and that when it has reached its verdict all the rough work done by $\mathfrak{D}$ is erased.

The machine $\mathcal{H}$ has its motion divided into sections. In the first $N-1$ sections, among other things, the integers 1, 2, ..., $N-1$ have been written down and tested by the machine $\mathfrak{D}$. A certain number, say $R(N-1)$, of them have been found to be the D.N's of circle-free machines. In the N-th section the machine $\mathfrak{D}$ tests the number N. If N is satisfactory, *i.e.*, if it is the D.N of a circle-free machine, then $R(N) = 1+R(N-1)$ and the first $R(N)$ figures of the sequence of which a D.N is N are calculated. The $R(N)$-th figure of this sequence is written down as one of the figures of the sequence β' computed by $\mathcal{H}$. If N is not satisfactory, then $R(N) = R(N-1)$ and the machine goes on to the $(N+1)$-th section of its motion.

From the construction of $\mathcal{H}$ we can see that $\mathcal{H}$ is circle-free. Each section of the motion of $\mathcal{H}$ comes to an end after a finite number of steps. For, by our assumption about $\mathfrak{D}$, the decision as to whether N is satisfactory is reached in a finite number of steps. If N is not satisfactory, then the N-th section is finished. If N is satisfactory, this means that the machine $\mathcal{M}(N)$ whose D.N is N is circle-free, and therefore its $R(N)$-th figure can be calculated in a finite number of steps. When this figure has been calculated and written down as the $R(N)$-th figure of β', the N-th section is finished. Hence $\mathcal{H}$ is circle-free.

Now let K be the D.N of $\mathcal{H}$. What does $\mathcal{H}$ do in the K-th section of its motion? It must test whether K is satisfactory, giving a verdict "s" or "u". Since K is the D.N of $\mathcal{H}$ and since $\mathcal{H}$ is circle-free, the verdict cannot be "u". On the other hand the verdict cannot be "s". For if it were, then in the K-th section of its motion $\mathcal{H}$ would be bound to compute the first $R(K-1)+1 = R(K)$ figures of the sequence computed by the machine with K as its D.N and to write down the $R(K)$-th as a figure of the sequence computed by $\mathcal{H}$. The computation of the first $R(K)-1$ figures would be carried out all right, but the instructions for calculating the $R(K)$-th would amount to "calculate the first $R(K)$ figures computed by H and write down the $R(K)$-th". This $R(K)$-th figure would never be found. *I.e.*, $\mathcal{H}$ is circular, contrary both to what we have found in the last paragraph and to the verdict "s". Thus both verdicts are impossible and we conclude that there can be no machine $\mathfrak{D}$.