A note on Türing 1936

Paola Cattabriga

September 26, 2022

A note on Türing's 1936*

Paola Cattabriga

University of Bologna

Abstract

Türing's article of 1936 claims to have defined a number which is not computable, arguing that there can be no machine computing the diagonal on the enumeration of the computable sequences. This article closely examines the original 1936 argument, displaying how it cannot be considered a demonstration, and that there is indeed no evidence of such a defined number that is not computable.

1 Introduction

As well known, Türing historical article of 1936 is the result of a special endeavor focused around the factuality of a general process for algorithmic computation. As resultant formal model his famous abstract computing machine, soon called Türing machine, could be regarded to be a universal feasibility test for computing procedures. The article begins by accurately outlining the notion of computable number, that is a real number is computable only if there exists a Türing machine that writes all the sequence of its decimal extension. The abstract machine as a universal feasibility test for computing procedures is then applied up to closely examining what are considered to be the limits of computation itself, and to defining a number which is not computable.

> The computable numbers do not include, however, all definable numbers; and an example is given of a definable number which is not computable (230 [1]).

In Section 8. is reached the crucial demonstration establishing some fundamental limits of computation by defining such number through a self-referring procedure. The present note shows how this procedure can not actually be regarded as a demonstration. In the following the reader is required to know Türing article together with the original notions and symbolism therein contained [1]. We recall briefly to the reader only a few of the main ones, with an example.

Computing machines. If any automatic machine $\mathcal{M}$ prints two kinds of symbols, of which the first kind 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 configuration, the subsequence of the symbols printed by it which are of the first kind will be called the *sequence computed by the machine.*

Circular and circle-free machines. If a computing machine $\mathcal{M}$ 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.

*Presented in Haifa at WIL2022 on Sunday, July 31st, this article expands and displays previous arXiv: 1308.0497.

Computable sequences. A sequence is said to be computable if it can be computed by a circle-free machine.

Computable numbers. A number is computable if it differs by an integer from the number computed by a circle-free machine.

S.D. Any automatic machine $\mathcal{M}$ is identified by its Table describing configurations and behaviors. Any Table can be coded or rewritten in a new description called the *Standard Description* of $\mathcal{M}$ (Example 1.1).

Example 1.1. *The table of m-configurations of a machine $\mathcal{M}$ computing the infinite sequence* 01010101 . . .

$$\begin{array}{llll} q_1 & S_0 & PS_1, R & q_2 \\ q_2 & S_0 & PS_0, R & q_3 \\ q_3 & S_0 & PS_2, R & q_4 \\ q_1 & S_0 & PS_0, R & q_1 \end{array}$$

which can be arranged on a line

$$q_1\, S_0\, PS_1, R\, q_2; \;\; q_2\; S_0\; PS_0, R\; q_3; \;\; q_3\; S_0\; PS_2, R\; q_4; \;\; q_1\; S_0\; PS_0, R\; q_1\; ; .$$

Standard Description of $\mathcal{M}$

$$DADDCRDAA; \;\; DAADDRDAAA; \;\; DAAADDCCRDAAAA; DAAAADDRDA; .$$

Description Number of $\mathcal{M}$

$$3133253117311335311173111332253111173111335317$$

D.N. Any letter in the standard description of $\mathcal{M}$ can be replaced by a number, so we shall have a description of the machine in the form of an arabic numeral. The integer represented by this numeral is called *Description Number.* A number which is a description number of a circle-free machine will be called a *satisfactory* number (Example 1.1).

Universal Machine. A universal machine is a computing machine $\mathcal{U}$ that, supplied with a tape on the beginning of which is written the *S.D.* of a computing machine $\mathcal{M}$, computes the same sequence of $\mathcal{M}$.

A simple representation to view the Universal Machine in modern terms:

$$\text{input } \; \underline{\textit{S.D.} \text{ of } \mathcal{M}} \longrightarrow \boxed{\mathcal{U}} \longrightarrow \text{output } \; \underline{\text{sequence computed by } \mathcal{M}}$$

2 The diagonal process

At the beginning of Section 8. *Application of the diagonal process.*, Türing intends to submit to his machine's feasibility test the application of Cantor's non-denumerability of real numbers to the computable sequences. He verifies so if the diagonal process is suitable to show also the non-denumerability of computable sequences.

> 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 (246 [1]).

A brief and elegant diagonalization is then proposed as follows:

$$
\begin{array}{lllll}
a_1 = & \phi_1(1) & \phi_1(2) & \phi_1(3) & \dots \\
 & & \ddots & & \\
a_2 = & \phi_2(1) & \phi_2(2) & \phi_2(3) & \dots \\
 & & & \ddots & \\
a_3 = & \phi_3(1) & \phi_3(2) & \phi_3(3) & \dots \\
\vdots & & & & \ddots \\
a_n = & \phi_n(1) & \phi_n(2) & \phi_n(3) & \dots \phi_n(n) \\
\vdots & & & & \qquad \ddots
\end{array}
$$

where a_n are the computable sequences with the figures $\phi_n(m)$ (on to $0, 1$), and β is the sequence with $1 - \phi_n(n)$ as its n-th figure.

> $\star$) Since β is computable, there exist 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. The computable sequences are therefore not enumerable.

Türing himself considers argument $\star$) fallacious as it presupposes the computability of β, which in turn presupposes the enumerability of computable sequences by finite means. For Türing the problem of enumerating computable sequences would be equivalent to finding out whether a given number is the *D.N* of a circle free machine and he seems certain that the feasibility test provided by his machine will show the impossibility for any such process. The most direct proof of impossibility could be to show that there exists a machine that computes β. Türing seems here attribute to the reader a special undefined incertitude, a feeling that "there must be something wrong". We will not dwell upon whether it should be the reader or it was Türing himself to have such inconvenient or inexplicable feeling[1].

So he chooses to test the feasibility of such a general process for finding whether a given number is the *D.N.* of a circle free machine, by means of a self referring argument. His argumentation will not be based on β, but on constructing β', whose n-th figure is $\phi_n(n)$, i.e. the same diagonal sequence $\phi_1(1)\phi_2(2)\phi_3(3)\dots\phi_n(n)\dots$.

[1] A display of the inferential steps in $\star$) offers perhaps some explicative insight about.

$$
\begin{array}{rcccccc}
 & \beta = 1-\phi_1(1) & 1-\phi_2(2) & 1-\phi_3(3) & \dots & 1-\phi_n(n) & \dots \\
1-\phi_n(n) = \phi_K(n) \text{ all } n & \downarrow & \downarrow & \downarrow & & \downarrow & \\
 & K = \ \phi_K(1) & \phi_K(2) & \phi_K(3) & \dots & \phi_K(n) & \dots \\
K = n & \downarrow & \downarrow & \downarrow & & \downarrow & \\
 & n = \ \phi_n(1) & \phi_n(2) & \phi_n(3) & \dots & \phi_n(n) & \dots
\end{array}
$$

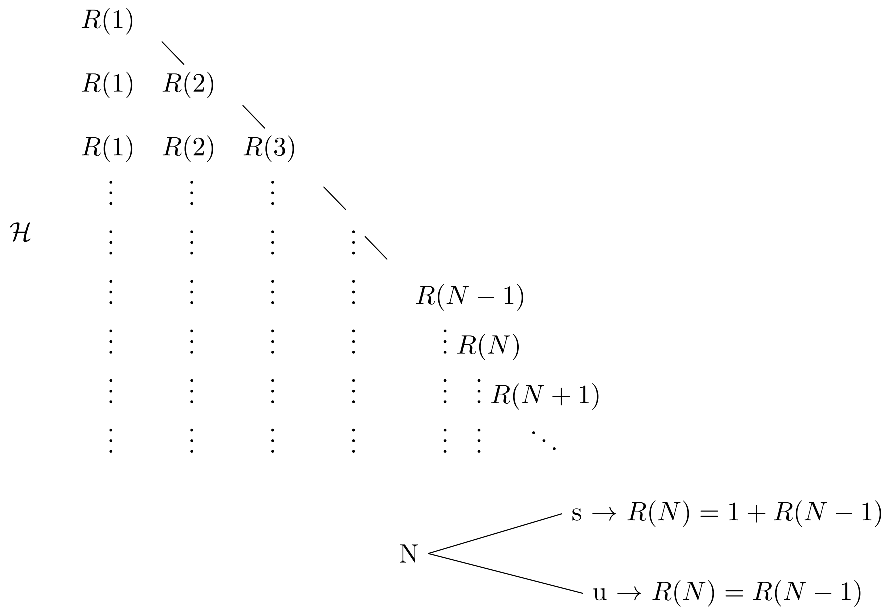

Figure 1: a representation of $\mathcal{H}$ computing the diagonal β'

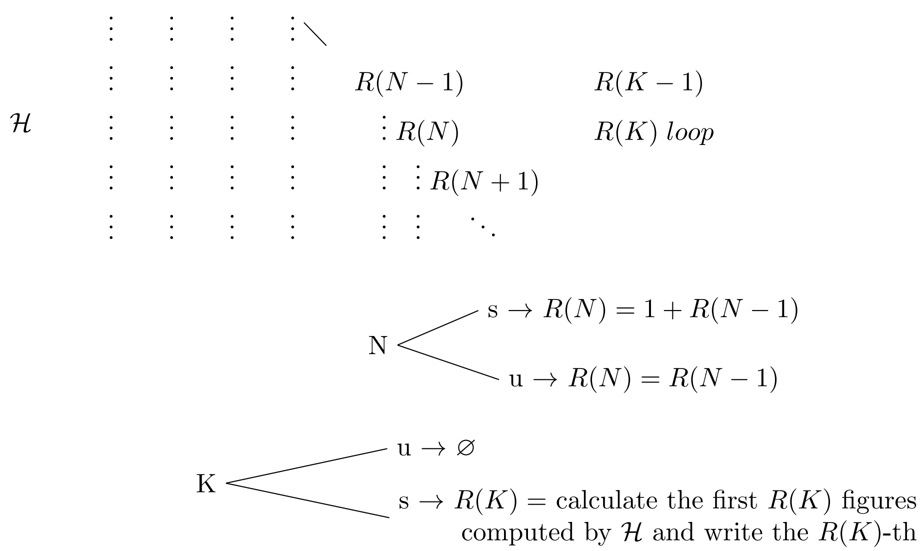

Figure 2: $\mathcal{H}$ encounters its own *D.N. K*

3 The main argument

The whole section 8 is based on the "proof" that it cannot exist an effective process constructing β', namely there is no feasible process generating $\phi_1(1)\phi_2(2)\phi_3(3)\dots\phi_n(n)\dots$.

Türing's "proof" is by reductio ad absurdum, assuming that such a process exists for real. That would be, we have a machine $\mathcal{D}$ that given the *S.D.* of any machine $\mathcal{M}$ will test if $\mathcal{M}$ is circular, marking the *S.D.* with "u", or is circle-free, marking the *S.D.* with "s".

input *S.D.* of $\mathcal{M}$ $\longrightarrow$ $\boxed{\mathcal{D}}$ $\longrightarrow$ output

or $\mathcal{M}$ is circular then mark *S.D.* with "u"

or $\mathcal{M}$ is not circular then mark *S.D.* with "s"

And then to construct a machine $\mathcal{H}$ by combining $\mathcal{D}$ and $\mathcal{U}$, where $\mathcal{U}$ simulates $\mathcal{M}$, and generates the computable sequence β'.

input *S.D.* of $\mathcal{M}$ $\longrightarrow$ $\boxed{\mathcal{D}}$ $\longrightarrow$ output

or $\mathcal{M}$ is circular then mark *S.D.* with "u"

or $\mathcal{M}$ is not circular then mark *S.D.* with "s" and

input *S.D.* of $\mathcal{M}$ $\longrightarrow$ $\boxed{\mathcal{U}}$ $\longrightarrow$ output computable sequence $\mathcal{M}$

The machine $\mathcal{H}$ would have its motion divided into sections as follows. In the first $N-1$ sections, among other things, the integers $1, 2, \ldots N-1$ will have been written down and tested by the machine $\mathcal{D}$. A certain number, say $R(N-1)$, of them will have been found to be the *D.N.*'s of circle-free machines. In the N-th section the machine $\mathcal{D}$ 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 (247 [1]) (Figure 1 on page 4).

The whole argument leads to contradiction when $\mathcal{H}$ encounters itself, namely its own *D.N.* K, turning out to be $\mathcal{H}$ in the meantime circular and circle-free (Figure 2 on page 4). 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 $\mathcal{H}$ and write down the $R(K)$-th". This $R(K)$-th figure would never be found.

But since $\mathcal{H}$ is, even if by theoretical assumption, a machine, it is an effective process, so $\mathcal{H}$ is associated to its Table of m-configurations, ad therefore to its *S.D.*. Accordingly the assumption of existence of $\mathcal{H}$ contains also that its *D.N.* K is known and very well coded, whole Türing's formalism was built to the purpose. The consequences of this on the plane of the machines themselves seem to be neglected in Türing's entire argument. Nothing really prevents us to define a machine $\mathcal{H}'$ that is the same as $\mathcal{H}$ except that if it encounters the *D.N.* K do not upload it in the $R(N)$-th figure of β'. So that if $\mathcal{H}$ is such that

$$R(N) \begin{cases} N = s & \text{then} \qquad 1 + R(N-1) \\ N = u & \text{then} \qquad R(N-1) \end{cases} \tag{1}$$

where the number $R(N)$ is the $R(N)$-th figure of β', generated by $\mathcal{H}$, then we can define $\mathcal{H}'$ with the instructions such that

$$R'(N) \begin{cases} N = K & \text{then} \qquad R'(N-1) \\ N = s & \text{then} \qquad 1 + R'(N-1) \\ N = u & \text{then} \qquad R'(N-1) \end{cases} \tag{2}$$

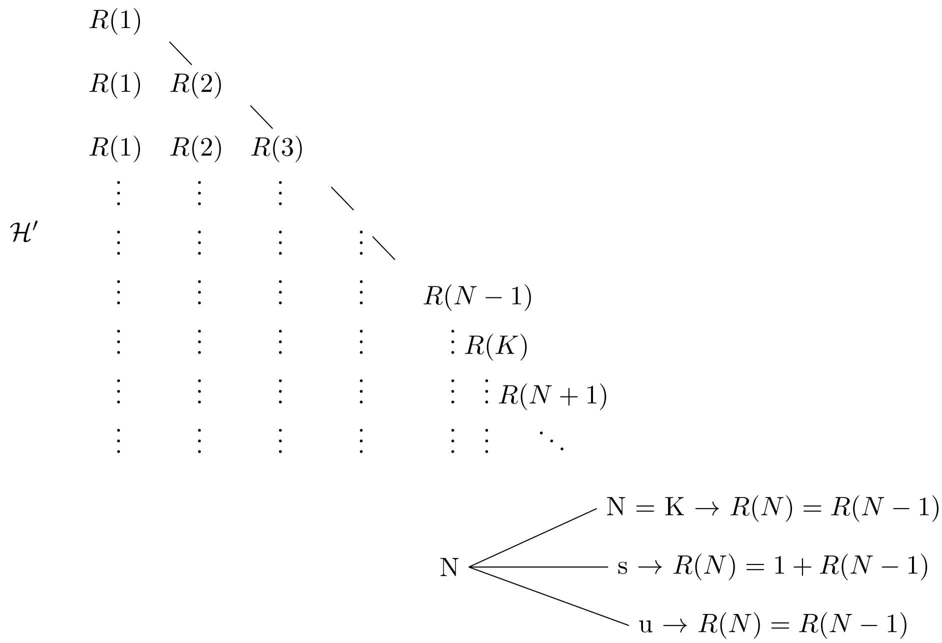

Figure 3: $\mathcal{H}'$ keep computing when encouters K

where $R'(N)$ is the $R(N)$-th figure of β' without $R(K)$. Actually when H' is in the N-th section such that $N = K$, $\mathcal{H}'$ goes on to the $(N+1)$-th section of its motion. So K, as well as the u numbers, is not included in the $R(N)$-th figure of β'. So what does $\mathcal{H}'$ do in the K-th section of $\mathcal{H}$? Simply $\mathcal{H}'$ goes on to the $(K+1)$-th section of $\mathcal{H}$, and its computation would be carried on (Figure 3 on page 6). We cannot then state that $\mathcal{H}'$ is circular like $\mathcal{H}$. When K is encountered, $\mathcal{H}$ stops but $\mathcal{H}'$ continues computing the computable sequences of $\mathcal{H}$. And $\mathcal{H}'$ is an effective process satisfying the feasibility test longed by Türing, which can always be defined whenever $\mathcal{H}$ is too, so his whole argument fails to reach a contradiction, and does not obtain a result of effective impossibility of the beginning assumption of the existence of $\mathcal{D}$.

We just have no conclusion that there can be no machine $\mathcal{D}$. There is therefore no proof about having no general process for finding out whether a given number is the *D.N.* of a circle-free machine. We can then regard accordingly all the other arguments arising (248, 259-265 [1]). Furthermore, considering (2), there is no evidence that K is not effectively computable, $\mathcal{H}'$ is indeed its computation, so there is not even an example of a definable number which is not computable. Let us observe that the notion of circle-free machines echoes a lot the requirement that a definition must not be circular, which is what in the Theory of Definition is known to be ruled by the Criterion of eliminability. A definition that does not satisfy this requirement introduces a primitive term indeed, and it is not a definition at all. One might object that the construction of the number K is not a definition, but this would not be as stated in [1].

References

[1] Türing, A., On Computable Numbers with an Application to the Entscheidungsproblem. *Proc. of the London Mathematical Society*, 42, 1936, pp. 230-67; e. c.43, 1937, pp.544-46.