



ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

- By A. M. TURING

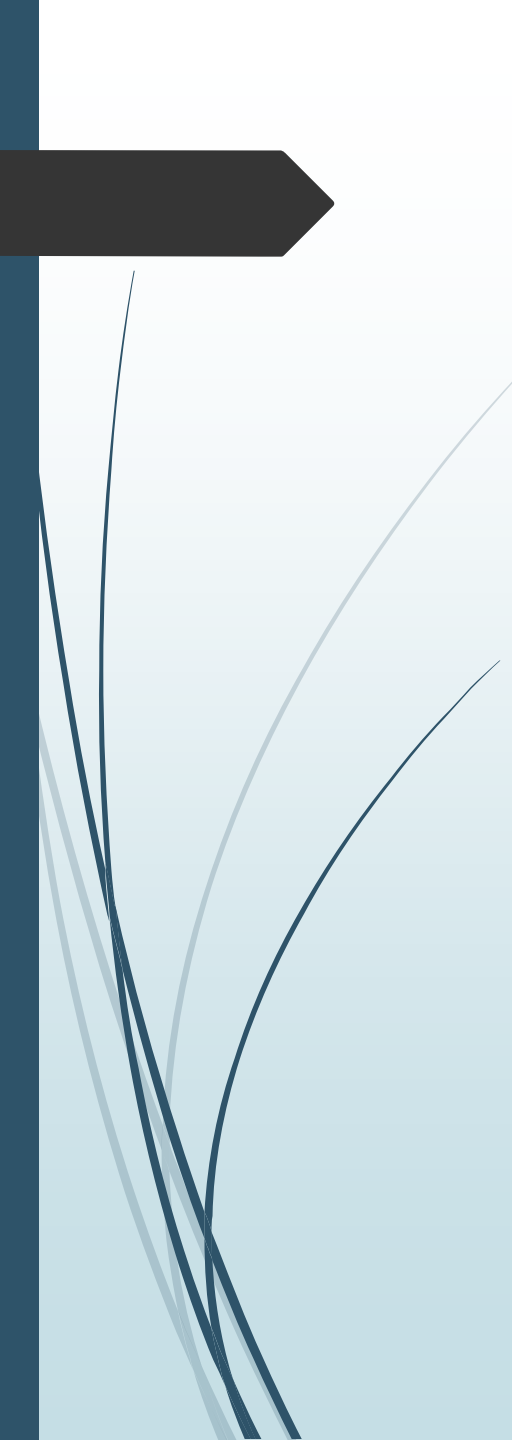
- 
- Em 1928, Hilbert apresentou um desafio à comunidade matemática - criar um algoritmo que toma como entrada uma afirmação na lógica de primeira ordem (possivelmente com um número finito de axiomas adicionais) e responde “Sim” ou “Não” se é universalmente válido.
 - Ter esse algoritmo significaria que não existe um problema insolúvel. Você pode construir um algoritmo para resolver qualquer problema que possa declarar usando instruções lógicas de primeira ordem com predicados.
 - Considere a seguinte afirmação: $\forall x(C(x) \rightarrow \neg D(x))$. Podemos ter um algoritmo que nos diz se esta afirmação é universalmente válida?
 - Para responder à pergunta, Turing teve que começar definindo formalmente a computabilidade e o algoritmo.

- 
- Os números “computáveis” podem ser brevemente descritos como os números reais cuja parte decimal pode ser calculada através de meios finitos.
 - Segundo a definição dada por Turing, um número será computável se o mesmo pode ser escrito por uma máquina.
 - O conjunto dos números computáveis engloba todos os números que poderiam ser considerados como computáveis (prova mais a frente).
 - Alguns exemplos são: a parte real dos números algébricos, os zeros das funções de Bessel, e , π . Porém, nem todos os números definidos são computáveis.
 - Mesmo que englobe uma grande quantidade de números (e mesmo que seja parecido com a classe dos números reais), o conjunto dos números computáveis é enumerável.



1 – Computing Machines

- Vamos supor que uma máquina irá computar um número real.
- Para que isso seja feito, ela precisará seguir uma quantidade finita de condições, q_1, q_2, \dots, q_k . Isto será chamado de “**m-configurações**”.
- Nossa máquina estará munida de uma fita, que percorrerá por ela. Esta fita está dividida em **seções (também chamados de quadrados)**, e cada seção é capaz de conter um “**símbolo**”.
- Em qualquer dado instante, só poderemos ter um quadrado “dentro” da máquina, e este quadrado possuirá um símbolo r , que também estará dentro da máquina.
- Este símbolo receberá o nome de “**símbolo escaneado**”, que será o único símbolo que a máquina “sabe que existe”.

- 
- Apesar disso, ao alterar sua m-configuração, a máquina poderá se lembrar de alguns símbolos que já foram escaneados.
 - De qualquer forma, o comportamento da máquina é determinado pela m-configuração q_n e o símbolo escaneado, r . Assim, o par q_n, r será chamado de “**configuração**”.
 - Em algumas configurações, a máquina apaga o símbolo escaneado e, em outras, ela escreve um novo. Ela poderá, também, mudar o quadrado que está sendo escaneado, mas só movendo ele uma casa para a esquerda ou direita. Além disso, a m-configuração poderá ser alterada para uma outra qualquer.
 - Alguns dos símbolos irão formar uma sequência de figuras que representam a parte decimal do nome que está sendo computado. Alguns servirão apenas para auxiliar o processo de computação. Somente os últimos que poderão ser apagados (por que, se queremos representar um número, não apagaríamos a parte deste número que já foi representada: gastaríamos mais tempo escrevendo-os novamente).




2 – Definitions

- **Máquinas automáticas –**
- Uma máquina será considerada automática (recebendo o nome de **a-máquina**) quando cada estágio de sua movimentação for determinado por sua configuração.
- Em alguns casos, precisaremos utilizar máquinas de escolha (recebem o nome de **c-máquinas**). Sua movimentação só é parcialmente determinada pela configuração. Ao se deparar com uma situação ambígua, um operador externo precisará fazer uma escolha que ela deve tomar. Para os nossos estudos, vamos considerar apenas as a-máquinas.



➤ Máquinas de Computação –

- Se uma a -máquina imprime dois tipos de símbolos, onde o primeiro tipo (chamado de figuras) consiste apenas de 0s e 1s (os outros são chamados de símbolos de tipo secundário), então essa máquina será chamada de **máquina de computação**.
- Ao ser munida de uma fita e for “ligada”, a sub-sequência de símbolos imprimidos que fazem parte do primeiro tipo receberão o nome de **sequência computada pela máquina**.
- O número real, expresso em binário, é obtida ao se fazer a conversão de binário para decimal. Ele recebe o nome de **número computado pela máquina**.
- Em qualquer estágio do movimento da máquina, o “número” de quadrados escaneados, a sequência de todos os símbolos já impressos na fita e a m -configuração atual recebe o nome de **configuração completa** do tal estado em que ela se encontra.

- 
- Obs.: Chamaremos as mudanças na máquina e fita entre configurações que foram realizadas com sucesso de **movimentos** da máquina.
 - **Máquinas circulares e máquinas círculo-livre –**
 - Uma máquina de computação é dita **circular** se não escrever mais que uma quantidade finita de símbolos do primeiro tipo. Caso contrário, ela é chamada de **livre de círculos**.
 - Assim, para que a máquina seja circular, ela deve chegar a uma configuração aonde não hajam mais movimentos possíveis ou, se ainda houverem movimentos, estes não podem imprimir símbolos da primeira ordem.



► Sequências computáveis e números –

- Uma sequência é computável se ela pode ser computada por uma máquina círculo-livre.
- Um número é computável se ele difere por um inteiro de um número computador por uma máquina círculo-livre.

3 – Examples of Computing Machines

- No primeiro exemplo, vamos construir uma máquina compute a sequência 010101....
- Nossa máquina terá quatro m-configurações: “b”, “c”, “f”, “e”; Ela será capaz de imprimir “0” e “1”.
- Daremos a seguinte legenda para representar o comportamento da máquina:
 - “R” indica que a máquina se move um quadrado para a direita e imediatamente escaneia o símbolo presente nele;
 - “L” tem a mesma função, só muda a direção para a esquerda;
 - “E” significa que o símbolo escaneado é apagado;
 - “P” significa “imprime”.

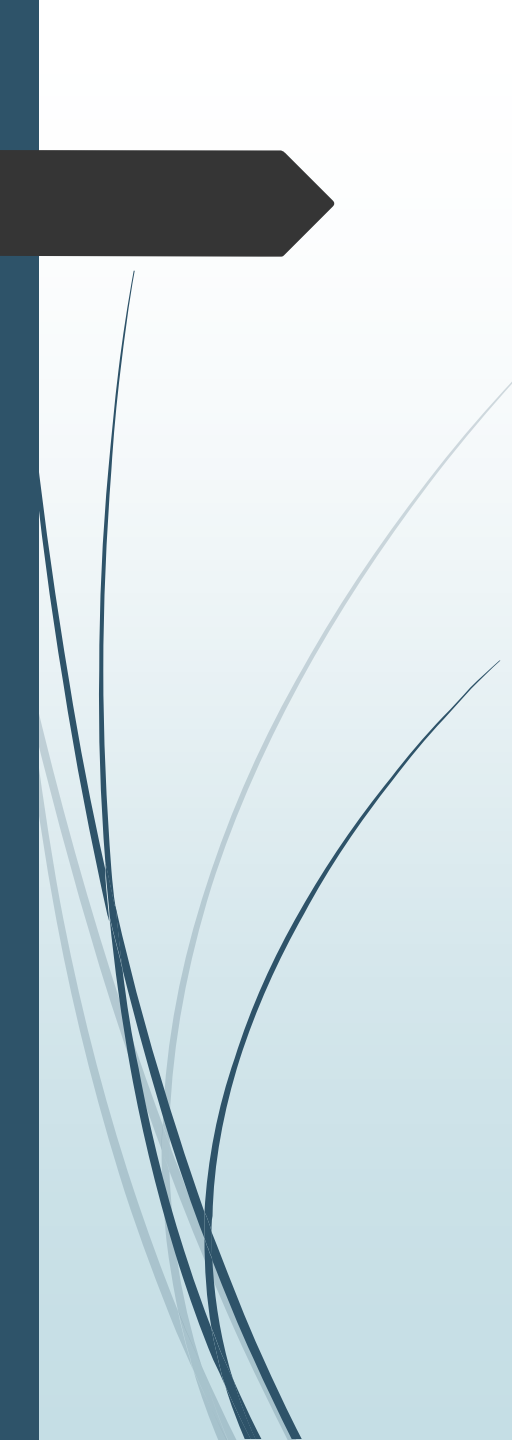
- Com estes símbolos e alguns outros, montaremos uma tabela que descreve o comportamento de uma máquina. Ela será interpretada da seguinte maneira:
- Para uma configuração descrita nas duas primeiras colunas, as operações da terceira coluna são executadas e, depois disso, a máquina vai para a m-configuração descrita na última coluna.
- Quando a segunda coluna é deixada em branco, o comportamento da terceira e da quarta coluna são aplicados para qualquer símbolo e para nenhum símbolo (?). Nossa máquina irá começar na m-configuração b com uma fita em branco.

<i>Configuration</i>		<i>Behaviour</i>		
<i>m-config.</i>	<i>symbol</i>	<i>operations</i>	<i>final</i>	<i>m-config.</i>
b	None	$P0, R$		c
c	None	R		c
e	None	$P1, R$		f
f	None	R		b

- Se permitirmos que as letras L e R apareçam mais de uma vez na coluna de operações, poderemos simplificar nossa tabela:

<i>m-config.</i>	<i>symbol</i>	<i>operations</i>	<i>final m-config.</i>
b	None	<i>P0</i>	b
	0	<i>R, R, P1</i>	b
	1	<i>R, R, P0</i>	b

- Vamos montar um exemplo mais complexo; Imagine que queremos montar uma máquina que compute a sequência 00101101110111101111....

- 
- Nossa máquina terá cinco m-configurações: “o”, “q”, “p”, “f”, “b”.
 - Ela será capaz de imprimir os seguintes símbolos: “ \emptyset ”, “x”, “0”, “1”.
 - Os três primeiros símbolos da nossa fita deverão ser “ $\emptyset\emptyset 0$ ”.
 - As próximas figuras seguirão em quadrados alternados.
 - Nos quadrados intermediários imprimiremos “x”. Ele servirá para nos organizar(original: keep the place) e, depois que terminarmos, os apagaremos.
 - Não poderemos ter quadrados em branco na sequência de figuras nos quadrados alternados.

<i>Configuration</i>		<i>Behaviour</i>	
<i>m-config.</i>	<i>symbol</i>	<i>operations</i>	<i>final m-config.</i>
b		$P\emptyset, R, P\emptyset, R, P0, R, R, P0, L, L$	c
c	$\left\{ \begin{array}{l} 1 \\ 0 \end{array} \right.$	R, Px, L, L, L	c
			q
q	$\left\{ \begin{array}{l} \text{Any (0 or 1)} \\ \text{None} \end{array} \right.$	R, R	q
		$P1, L$	p
p	$\left\{ \begin{array}{l} x \\ \emptyset \\ \text{None} \end{array} \right.$	E, R	q
		R	f
		L, L	p
f	$\left\{ \begin{array}{l} \text{Any} \\ \text{None} \end{array} \right.$	R, R	f
		$P0, L, L$	c

- A seguir, daremos uma ilustração dessa máquina em ação. Algumas configurações completas serão mostradas.
- Estas configurações são descritas anotando a sequência de símbolos que estão na fita, com a m-configuração escrita abaixo do símbolo escaneado. As configurações sucessivas completas estão separadas por dois pontos (:).

```

      : a a 0   0 : a a 0   0 : a a 0   0 : a a 0   0   : a a 0   0   1 :
b       a           q           q           q .           p
a a 0   0   1 : a a 0   0   1 : a a 0   0   1 : a a 0   0   1 :
      p           p           f           f
a a 0   0   1 : a a 0   0   1   : a a 0   0   1   0 :
      f           f           a
a a 0   0   1 x 0 : ....
      a

```

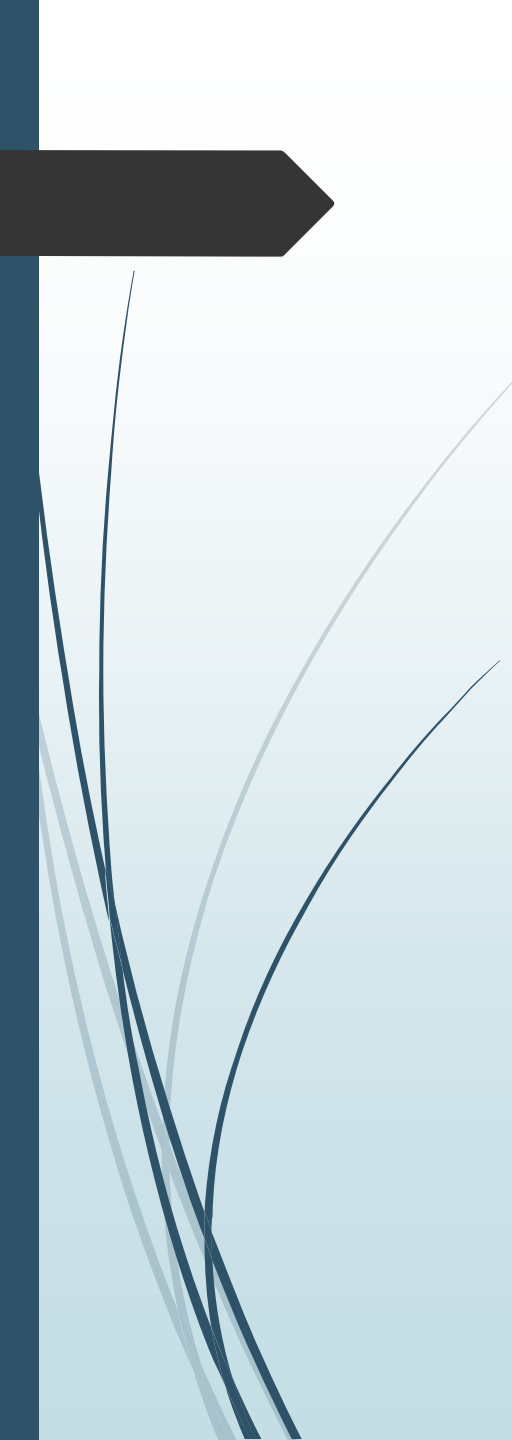
- Esta mesma tabela poderia ser expressa da seguinte maneira:

$\beta : \alpha \alpha \alpha 0 \quad 0 : \alpha \alpha \alpha 0 \quad 0 : \dots$

- A m-configuração foi adicionada a esquerda do símbolo escaneado. Apesar de ser mais complicada, esta forma será usada mais tarde.
- A sequência de quadrados que possuem símbolos escritos sobre eles serão chamados de F-Quadrados, enquanto a sequência de quadrados que estão em branco serão chamados de E-Quadrados.
- Os símbolos nos F-Quadrados formam uma sequência contínua. Não haverá nenhum f-quadrado em branco até que a sequência acabe.
- Os símbolos nos E-Quadrados poderão ser apagados. Só temos um e-quadrado entre dois f-quadrados.
- Se um símbolo β está impresso num F-Quadrado S e um símbolo α está impresso num E-quadrado a direita de S , então **S e β são marcados com α .**
-

4 – Abbreviated tables.

- Como existem certos processos que são basicamente utilizados por todas as máquinas (comparar sequências, apagar todos os símbolos de um dado tipo, copiar uma sequência de símbolos, etc.), utilizaremos, quando conveniente, as “**tabelas esqueleto**”, a fim de abreviar as tabelas das m-configurações consideravelmente.
- Nas tabelas esqueleto, aparecerão letras maiúsculas germânicas e letras minúsculas gregas. Estas terão a mesma natureza que as variáveis.
- Ao substituir cada letra germânica por uma m-configuração e cada letra grega por um símbolo, obteremos a tabela para uma m-configuração.
- Estas tabelas servem apenas para abreviação! Elas não são essenciais.


- 
- Se substituíssemos \mathbb{C} por q , \mathbb{B} por r , a por x , então teríamos a tabela completa para a m -configuração $f(q, r, x)$. f é chamado de “**função m -configuração**”, ou simplesmente “ **m -função**”.
 - As únicas expressões que podem ser utilizadas para a substituição em uma m -função são m -configurações e símbolos da máquina. Elas devem ser enumeradas, ainda mais se substituirmos uma m -função dentro da m -função.
 - Se não fizéssemos a enumeração, muito provavelmente teríamos infinitas m -configurações. Um exemplo disto é quando dizemos que nossa máquina deve ter a m -configuração q , e todas as outras m -configurações são obtidas por substituir uma m -configuração por C em $p(C)$. Assim, teríamos: $q, p(q), p(p(q)), p(p(p(q))), \dots$ como m -configurações.
 - Resumidamente, teremos as m -configurações da máquina (provavelmente expressas em termos de m -funções) e teremos as tabelas esqueleto. Para obter a tabela original, basta fazermos repetidas substituições.

- Aqui estão alguns exemplos:

(In the explanations the symbol “ \rightarrow ” is used to signify “the machine goes into the m -configuration. . . .”)

$e(\mathbb{C}, \mathfrak{B}, a)$	$f(e_1(\mathbb{C}, \mathfrak{B}, a), \mathfrak{B}, a)$	From $e(\mathbb{C}, \mathfrak{B}, a)$ the first a is
$c_1(\mathbb{C}, \mathfrak{B}, a)$	E	erased and $\rightarrow \mathbb{C}$. If there is no
	\mathbb{C}	$a \rightarrow \mathfrak{B}$.
$e(\mathfrak{B}, a)$	$c(e(\mathfrak{B}, a), \mathfrak{B}, a)$	From $c(\mathfrak{B}, a)$ all letters a are
		erased and $\rightarrow \mathfrak{B}$.

- Como o último exemplo pode ser um pouco mais complicado, imaginemos que $e(b, x) = q$. Assim, a tabela será:



$c(b, x)$

q

$e(c(b, x), b, x)$

$e(q, b, x).$

Or, in greater detail:

q

$c(q, b, x)$

$c(q, b, x)$

$f(e_1(q, b, x), b, x)$

$e_1(q, b, x)$

E

$q.$

- Aqui está um outro exemplo:

$pc(\mathbb{C}, \beta)$		$f(pc_1(\mathbb{C}, \beta), \mathbb{C}, \alpha)$	From $pc(\mathbb{C}, \beta)$ the machine prints β at the end of the sequence of symbols and $\rightarrow \mathbb{C}$.
$pc_1(\mathbb{C}, \beta)$	$\begin{cases} \text{Any} & R, R \\ \text{None} & P\beta \end{cases}$	$pc_1(\mathbb{C}, \beta)$	
		\mathbb{C}	
$l(\mathbb{C})$	L	\mathbb{C}	From $f'(\mathbb{C}, \mathfrak{B}, \alpha)$ it does the same as for $f(\mathbb{C}, \mathfrak{B}, \alpha)$ but moves to the left before $\rightarrow \mathbb{C}$.
$r(\mathbb{C})$	R	\mathbb{C}	
$f'(\mathbb{C}, \mathfrak{B}, \alpha)$		$f(l(\mathbb{C}), \mathfrak{B}, \alpha)$	
$f''(\mathbb{C}, \mathfrak{B}, \alpha)$		$f(r(\mathbb{C}), \mathfrak{B}, \alpha)$	
$c(\mathbb{C}, \mathfrak{B}, \alpha)$		$f'(c_1(\mathbb{C}), \mathfrak{B}, \alpha)$	$c(\mathbb{C}, \mathfrak{B}, \alpha)$. The machine writes at the end the first symbol marked α and $\rightarrow \mathbb{C}$.
$c_1(\mathbb{C})$	β	$pc(\mathbb{C}, \beta)$	

- A última linha representa a quantidade de linhas obtidas por se substituir beta por qualquer outro símbolo na fita da máquina em questão.

► Outro exemplo:

$cc(\mathbb{C}, \mathfrak{B}, a)$

$c\left(c(\mathbb{C}, \mathfrak{B}, a), \mathfrak{B}, a\right)$

$cc(\mathfrak{B}, a)$

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

$rc(\mathbb{C}, \mathfrak{B}, a, \beta)$

$f\left(re_1(\mathbb{C}, \mathfrak{B}, a, \beta), \mathfrak{B}, a\right)$

$re_1(\mathbb{C}, \mathfrak{B}, a, \beta) \quad E, P\beta$

\mathbb{C}

$rc(\mathfrak{B}, a, \beta)$

$rc\left(rc(\mathfrak{B}, a, \beta), \mathfrak{B}, a, \beta\right)$

$cr(\mathbb{C}, \mathfrak{B}, a)$

$c\left(rc(\mathbb{C}, \mathfrak{B}, a, a), \mathfrak{B}, a\right)$

$cr(\mathfrak{B}, a)$

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

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

$rc(\mathbb{C}, \mathfrak{B}, a, \beta)$. The machine replaces the first a by β and $\rightarrow \mathbb{C} \rightarrow \mathfrak{B}$ if there is no a . $rc(\mathfrak{B}, a, \beta)$. The machine replaces all letters a by β ; $\rightarrow \mathfrak{B}$.

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

$$c(\mathbb{C}, \mathbb{A}, \mathbb{E}, \alpha, \beta) \quad \tilde{f}'(\text{cp}_1(\mathbb{C}_1 \mathbb{A}, \beta), f(\mathbb{A}, \mathbb{E}, \beta), \alpha)$$

$$\text{cp}_1(\mathbb{C}, \mathbb{A}, \beta) \quad \gamma \quad f'(\text{cp}_2(\mathbb{C}, \mathbb{A}, \gamma), \mathbb{A}, \beta)$$

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

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

$$\text{cpc}(\mathbb{C}, \mathbb{A}, \mathbb{E}, \alpha, \beta) \quad \text{cp}\left(c(e(\mathbb{C}, \mathbb{C}, \beta), \mathbb{C}, \alpha), \mathbb{A}, \mathbb{E}, \alpha, \beta\right)$$

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

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

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

$q(\mathfrak{E})$	$\begin{cases} \text{Any} \\ \text{None} \end{cases}$	R	$q(\mathfrak{E})$
$q_1(\mathfrak{E})$	$\begin{cases} \text{Any} \\ \text{None} \end{cases}$	R	$q_1(\mathfrak{E})$
$q(\mathfrak{E}, a)$			$q(\mathfrak{E})$
$q_1(\mathfrak{E}, a)$	$\begin{cases} a \\ \text{not } a \end{cases}$	L	\mathfrak{E}
			$q(q_1(\mathfrak{E}, a))$

$q(\mathfrak{E}, a)$. The machine finds the last symbol of form a . $\rightarrow \mathfrak{E}$.

$pe_2(\mathfrak{E}, a, \beta)$			$pe(pe(\mathfrak{E}, \beta), a)$
$ce_2(\mathfrak{B}, a, \beta)$			$ce(ce(\mathfrak{B}, \beta), a)$
$ce_3(\mathfrak{B}, a, \beta, \gamma)$			$ce(ce_2(\mathfrak{B}, \beta, \gamma), a)$

$pe_2(\mathfrak{E}, a, \beta)$. The machine prints $a \beta$ at the end.

$ce_3(\mathfrak{B}, a, \beta, \gamma)$. The machine copies down at the end first the symbols marked a , then those marked β , and finally those marked γ ; it erases the symbols a, β, γ .

► Continuação do último exemplo:

$e(\mathfrak{E})$	$\left\{ \begin{array}{ll} \mathfrak{o} & R \\ \text{Not } \mathfrak{o} & L \end{array} \right.$	$e_1(\mathfrak{E})$	From $e(\mathfrak{E})$ the marks are erased from all marked sym- bols. $\rightarrow \mathfrak{E}$.
$e_1(\mathfrak{E})$	$\left\{ \begin{array}{ll} \text{Any} & R, E, R \\ \text{None} & \end{array} \right.$	\mathfrak{E}	

5 – Enumeration of computable sequences

- Uma sequência computável γ é determinada **pela descrição da máquina** que computa γ .
- Dessa forma, a sequência 001011011101111... é determinada pela tabela que vimos anteriormente e, qualquer sequência computável pode ser descrita em termos da mesma máquina.
- Para nossa facilitar, padronizar estas tabelas irá nos facilitar bastante.
- A coluna de operações só poderá receber os seguintes comandos: E; E, R; E, L; Pa; Pa, L; Pa, R; R; L.
- Iremos enumerar nossas m -configurações, chamando-as de q_1, \dots, q_r . Nós também daremos números aos símbolos S_1, \dots, S_m . Em particular, o “vazio” = S_0 , “0” = S_1 e “1” = S_2 . Assim, as linhas da nossa tabela terão a seguinte forma:

<i>m</i> -config.	Symbol	Operations	<i>Final</i> <i>m</i> -config.	
q_i	S_j	$P\check{S}_k, L$	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

$$q_i \qquad S_j \qquad E, R \qquad q_m$$

are to be written as

$$q_i \qquad S_j \qquad PS_0, R \qquad q_m$$

and lines such as

$$q_i \qquad S_j \qquad R \qquad q_m$$

to be written as

$$q_i \qquad S_j \qquad PS_j, R \qquad q_m$$


- Com esta simbologia, podemos reduzir as linhas de nossa tabela a linhas no formato de N1, N2 e N3.
- Com cada uma dessas linhas, poderemos formar as seguintes expressões:

- Para N1: $q_i S_j S_k L q_m$

- Para N2: $q_i S_j S_k R q_m$

- Para N3: $q_i S_j S_k N q_m$.

- Tendo em vista isso, uma maneira de obter a descrição do que nossa máquina faz seria colocar todas as linhas de sua tabela na “mesma linha” e separá-las por ponto e vírgula.
- Nesta descrição, vamos também alterar a letra q_i por “D”, seguido pela letra “A” repetida i vezes. S_j será substituído por “D”, seguido por “C” repetido j vezes.
- Essa nova descrição da máquina receberá o nome de **descrição padrão (S.D)**. Ela é composta inteiramente pelas letras “A”, “C”, “D”, “L”, “R”, “N” e por “;”.

- 
- Poderíamos ir um pouco além e substituir “A” por “1”, “C” por “2”, “D” por “3”, “L” por “4”, “R” por “5”, “N” por “6” e “;” por 7. Dessa forma, teremos a descrição da máquina montada por algarismos. Esse número será chamado de **descrição numérica (D.N)** da máquina.
 - A D.N. determina o S.D., que por sua vez determina a estrutura da máquina.
 - Uma máquina que tem D.N. n pode ser descrita como $M(n)$.
 - Para cada sequência computável, existem pelo menos uma descrição numérica.
 - Uma descrição numérica corresponde a somente uma sequência computável. Dessa forma, as sequência computáveis e as descrições numéricas são **enumeráveis**.

- Vamos achar a descrição numérica de uma dada máquina.

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

- Poderíamos adicionar linhas irrelevantes a tabela para derivar outras. Um exemplo seria:

q_1	S_1	PS_1, R	q_2
-------	-------	-----------	-------

- Nossa forma padrão seria:

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

- A nossa descrição padrão será:

DADDCRDAA;DAADDRDAAA;

DAAADDCCRDAAAA;DAAAADDRDA;

- O nosso número de descrição será:

31332531173113353111731113322531111731111335317

- Da mesma maneira que é o seguinte número:

3133253117311335311173111332253111173111133531731323253117

- Um **Número Satisfatório** é o número de descrição de uma máquina círculo-livre. Mais a frente, mostraremos que não há um processo geral para se determinar se um dado número é satisfatório ou não.

6 – The Universal Computing Machine

- A máquina universal é uma máquina que pode calcular qualquer sequência computável. Quando dado como entrada alguma descrição padrão de uma máquina qualquer M , a máquina universal $U(M)$ calculará exatamente a mesma sequência que M .
- Como uma máquina é capaz de escrever qualquer símbolo em uma fita, somos capazes de construir uma máquina M' que escreva a descrição numérica (ou descrição padrão) de outra máquina de computação M . A partir disso, estas instruções podem ser utilizadas para calcular a sequência computada por M .
- Para que esta máquina se pareça ainda mais com U , bastaria adicionar uma operação que sobrescreve a descrição de M por uma descrição de outra máquina J .

7 - Detailed description of the universal machine

- Quando U está pronto para começar a trabalhar, a fita que está dentro dele contém o símbolo θ em um F-Quadrado e outro θ no próximo E-Quadrado. Depois disso, apenas em F-quadrados, vem a S.D. da máquina seguida por dois ponto e vírgulas “::” (um símbolo só). A partir disso, a S.D. consiste de várias instruções, separadas por dois pontos.
- Cada instrução consiste de cinco partes consecutivas:
- (i) “D” seguido por uma sequência de letras “A”. Isto descreve a m-configuração atual;
- (ii) “D”, seguido por uma sequência de letras “C”. Isto descreve o símbolo escaneado.
- (iii) “D”, seguido por outra sequência de letras “C”. Isto descreve qual símbolo no símbolo escaneado deve ser mudado.
- (iv) “L”, “R” ou “N”, descrevendo o próximo movimento da máquina.
- (v) “D”, seguido por uma sequência de letras A. Isto descreve a última m-configuração.

- U deve ser capaz de imprimir "A", "C", "D", "0", "1", "u", "v", "w", "x", "y", "z". A S.D. é formada por ";", "A", "C", "D", "L", "R", "N".
- Aqui estão suas tabelas:

Subsidiary skeleton table.

$\text{con}(\mathfrak{E}, a)$	$\left\{ \begin{array}{ll} \text{Not } A & R, R \quad \text{con}(\mathfrak{E}, a) \\ A & L, Pa, R \quad \text{con}_1(\mathfrak{E}, a) \end{array} \right.$	<p>$\text{con}(\mathfrak{E}, a)$. 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 a. $\rightarrow \mathfrak{E}$.</p>
$\text{con}_1(\mathfrak{E}, a)$	$\left\{ \begin{array}{ll} A & R, Pa, R \quad \text{con}_1(\mathfrak{E}, a) \\ D & R, Pa, R \quad \text{con}_2(\mathfrak{E}, a) \end{array} \right.$	
$\text{con}_2(\mathfrak{E}, a)$	$\left\{ \begin{array}{ll} C & R, Pa, R \quad \text{con}_2(\mathfrak{E}, a) \\ \text{Not } C & R, R \quad \mathfrak{E} \end{array} \right.$	<p>$\text{con}(\mathfrak{E},)$. 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.</p>

The table for \mathcal{U} .

b	$f(b_1, b_1, ::)$
b_1	$R, R, P :, R, R, PD, R, R, PA \quad \text{anf}$
anf	$g(\text{anf}_1, :)$
anf_1	$\text{con}(\text{fom}, y)$
fom	$\left\{ \begin{array}{ll} ; & R, Pz, L \\ z & L, L \\ \text{not } z \text{ nor } ; & L \end{array} \right. \begin{array}{l} \text{con}(\text{fmp}, x) \\ \text{fom} \\ \text{fom} \end{array}$
fmp	$\text{cpe}(c(\text{fom}, x, y), \text{sim}, x, y)$

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

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

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 .

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

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 . $\rightarrow \S \text{im}$.

$\S \text{im}$ $f'(\S \text{im}_1, \S \text{im}_1, z)$

$\S \text{im}_1$ $\text{con}(\S \text{im}_2,)$

$\S \text{im}_2 \left\{ \begin{array}{ll} A & \S \text{im}_3 \\ \text{not } A & R, Pu, R, R, R \end{array} \right. \S \text{im}_2$

$\S \text{im}_3 \left\{ \begin{array}{ll} \text{not } A & L, Py \\ A & L, Py, R, R, R \end{array} \right. e(mf, z) \S \text{im}_3$

mf $g(mf, :)$

$mf_1 \left\{ \begin{array}{ll} \text{not } A & R, R \\ A & L, L, L, L \end{array} \right. \begin{array}{l} mf_1 \\ mf_2 \end{array}$

$mf_2 \left\{ \begin{array}{ll} C & R, Px, L, L, L \\ : & \\ D & R, Px, L, L, L \end{array} \right. \begin{array}{l} mf_2 \\ mf_4 \\ mf_3 \end{array}$

$mf_3 \left\{ \begin{array}{ll} \text{not } : & R, Pv, L, L, L \\ : & \end{array} \right. \begin{array}{l} mf_3 \\ mf_4 \end{array}$

$\S \text{im}$. 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.

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. $\rightarrow \S h$.

mf_4 $con(l(l(mf_5)),)$

mf_5 $\begin{cases} \text{Any} & R, Pw, R \\ \text{None} & P: \end{cases}$ mf_5 sh

sh $f(sh_1, inst, u)$

sh_1 L, L, L sh_2

sh_2 $\begin{cases} D \\ \text{not } D \end{cases}$ R, R, R, R sh_2 $inst$

sh_3 $\begin{cases} C \\ \text{not } C \end{cases}$ R, R sh_4 $inst$

sh_4 $\begin{cases} C \\ \text{not } C \end{cases}$ R, R sh_5 $pe_2(inst, 0, :)$

sh_5 $\begin{cases} C \\ \text{not } C \end{cases}$ $inst$ $pe_2(inst, 1, :)$


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.


$inst$		$g(l(inst_1), u)$
$inst_1$	a	$R, E \quad inst_1(a)$
$inst_1(L)$		$ce_5(ov, v, y, x, u, w)$
$inst_1(R)$		$ce_5(ov, v, x, u, y, w)$
$inst_1(N)$		$cc_5(ov, v, x, y, u, w)$
ov		$e(anf)$

$inst$. 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

- Pode-se pensar que argumentos que provam que os números reais não são enumeráveis também provariam que os números computáveis e sequências não podem ser enumeradas.
- O problema de enumerar sequências computáveis é equivalente ao problema de descobrir se um determinado número é o D.N de um máquina livre de círculo, e não temos nenhum processo geral para fazer isso em um número finito de etapas.
- Turing usa o processo de diagonalização para mostrar que não podemos construir um processo que tome a descrição padrão de uma máquina e nos diga se ela é livre de círculos ou não. Isso é conhecido como o **Problema da Parada**.
- Suponha que podemos construir um máquina D que receberá uma descrição padrão. Esta descrição será marcada com u, se representar uma máquina circular, e com s se representar uma máquina livre de círculos. Se tomarmos D e U (máquina universal), podemos criar um máquina N que computa a sequência β , cuja enésima figura é $\varphi_n(n)$.


- 
- N primeiro avalia os números inteiros (descrição numérica) até $N-1$ e os testa com D . Até o momento, todos parecem ser números de descrição de máquinas livres de círculo. Finalmente, a máquina testa o número N . Se é o número de descrição de uma máquina sem círculo, então $R(N)$ é igual a $1 + R(N-1)$ e os primeiros números $R(N)$ da sequência para N são testados e escritos como partes da sequência β . Caso contrário, a máquina vai para o passo $N + 1$.
 - Podemos ver que N é livre de círculo - cada seção de β é calculada em passos finitos, porque assumimos que D é computável em um número finito de passos.
 - Mas digamos que K é o número de descrição de N . Sabemos que K é satisfatório, pois N é livre de círculos, mas para N verificar isso ele teria que computar os primeiros $R(K-1) + 1$ símbolos, e, para a $R(K)$ figura, ele teria que repetir tudo de novo.
 - Assim, N fica preso em um loop e chegamos a um paradoxo. Portanto, não pode ter tal máquina D .

- 
- Isso pode ainda ser expandido para mostrar que não pode haver tal máquina que determine se uma determinada máquina irá imprimir um símbolo em particular. Até que o símbolo apareça, como determinar se a máquina está presa em um loop ou se devemos esperar um pouco mais?



9. The extent of the computable numbers

- Qualquer prova de que números computáveis, conforme definido no artigo de Turing, abrangem todos os números naturalmente considerados computáveis, se resumiria a um apelo à intuição, o que não é matematicamente interessante, de modo que Turing apenas assume que esse é o caso.
- Assumindo isso, uma das consequências mais interessantes é que, se houver um processo geral para provar que uma função no cálculo de Hilbert é demonstrável, ela pode ser executada por uma máquina. É isso que Turing está construindo em direção a este artigo inteiro: provar o Entscheidungsproblem.
- Haverá três tipos de argumentos:
 - a) um apelo direto à intuição
 - b) uma prova da equivalência de duas definições
 - c) dando exemplos de grandes classes de números computáveis

- 
- **A)** A computação é normalmente feita escrevendo símbolos no papel. Podemos supor que este artigo é dividido em quadrados. Vamos imaginar que a computação em uma fita unidimensional e que só podemos usar um número finito de símbolos antes de ficar sem espaço em cada quadrado individual.
 - Além disso, pode-se supor que, em qualquer estágio da computação, o próximo passo é totalmente determinado pela sequência de símbolos no papel e pelo estado do computador. Como o tamanho de nossa máquina é limitado, o número de estados deve ser finito, mas sempre podemos substituir os estados ausentes por escrever mais símbolos no papel.
 - Portanto, é possível simplificar qualquer processo de cálculo até as unidades mais básicas que são indivisíveis:
 - a) mudando o símbolo em um dos quadrados observados
 - b) mudar o símbolo de um quadrado para o símbolo de um quadrado diferente
 - Qualquer um destes pode envolver a mudança de estado.
 - Tal máquina pode obviamente ser construída e terá um desempenho semelhante às máquinas de computação definidas anteriormente.

➡ B) WIP

