

# Capítulo 7

## Máquina de Turing

A Máquina de Turing é universalmente conhecida e aceita como a formalização de um algoritmo. Este mecanismo se assemelha em muito aos computadores atuais, embora tenha sido proposta por Alan Turing <sup>1</sup> em 1936, muito antes dos primeiros computadores digitais. A Máquina de Turing possui o mesmo poder computacional de qualquer outro computador de propósito geral.

### 7.1 Máquina Clássica

Este mecanismo é uma evolução da máquina de estados finita, seja ela um DFA ou um NFA. A diferença dela para os autômatos estudados até o presente momento é que a Máquina de Turing pode não só ler uma fita de entrada, mas também pode escrever na mesma. A fita de entrada é limitada à esquerda por um símbolo  $\triangleright$  marcador de fim de fita, enquanto que se estende indefinidamente à direita. Sempre

---

<sup>1</sup>Alan Mathison Turing, Ph.D. (1912-1954), inglês nascido no bairro de Paddington em Londres, é considerado um dos fundadores da ciência da computação. Durante a Segunda Guerra Mundial foi o responsável pela quebra da máquina de encriptação alemã chamada Enigma, contribuindo de forma significativa para o desfecho da guerra. Trabalhou com mecânica quântica, probabilidade, lógica, computabilidade, eletrônica, inteligência artificial, redes neurais e biologia. Foi cruelmente massacrado, preso e humilhado pela sociedade inglesa que o impedia de trabalhar devido a sua opção sexual. Um dos maiores vultos da computação suicidou-se em sua casa com uma maçã envenenada com cianeto. Atualmente, a *Association for Computing Machinery*, ou ACM, concede anualmente o prêmio Nobel da Computação, conhecido merecidamente como Prêmio Turing.

que o símbolo  $\triangleright$  é lido, a máquina move a cabeça leitora obrigatoriamente para a direita. Estes movimentos da cabeça leitora para a direita e esquerda são representados respectivamente por  $\rightarrow$  e  $\leftarrow$ . Além disto, a fita de entrada pode não ter todos os seus campos preenchidos com símbolos, isto é, ela pode ter espaços em branco. Um espaço em branco, por conveniência, é denotado como sendo o símbolo  $\sqcup$ .

**Definição 7.1** Uma Máquina de Turing é uma quintupla  $M = \langle k, \Sigma, \delta, s, F \rangle$  onde:

- $k$  é o conjunto finito de ESTADOS;
- $\Sigma$  é um ALFABETO que contém os símbolos  $\triangleright$  e  $\sqcup$ , mas não contém  $\rightarrow$  e  $\leftarrow$ ;
- $s \in k$  é o ESTADO INICIAL;
- $F \subseteq k$  é o conjunto de ESTADOS DE PARADA;
- $\delta$  é uma FUNÇÃO DE TRANSIÇÃO de  $k \times \Sigma$  onde:
  - (a) para todos os  $q \in (k - F)$ , se  $\delta(q, \triangleright) = (p, b)$ , então  $b = \rightarrow$ ;
  - (b) para todos os  $q \in (k - F)$  e  $a \in \Sigma$ ,  $a \neq \triangleright$ , se  $\delta(q, a) = (p, b)$ , então  $b \neq \triangleright$ .

**Definição 7.2** Seja  $M$  uma Máquina de Turing. Dizemos que a *configuração*  $C_1$  resulta em  $C_2$  se  $C_1 \vdash_M C_2$ . Uma *computação* por  $M$  é uma seqüência de configurações  $C_0, C_1, \dots, C_n$ , para algum  $n \geq 0$ , tal que  $C_0 \vdash_M C_1 \vdash_M \dots \vdash_M C_n$  e dizemos que a computação é de comprimento  $n$ , ou que possui  $n$  *passos*. Neste caso, escrevemos  $C_0 \vdash_M^n C_n$ .

Para entender seu funcionamento, considere a Máquina de Turing  $M = \langle k, \Sigma, \delta, s, F \rangle$ , onde  $k = \{q_0, q_1, h\}$ ,  $\Sigma = \{a, \sqcup, \triangleright\}$ ,  $s = q_0$ ,  $F = \{h\}$  e  $\delta$  é dada conforme a seguir:

$q$	$\sigma$	$\delta(q, \sigma)$
$q_0$	$a$	$(q_1, \sqcup)$
$q_0$	$\sqcup$	$(h, \sqcup)$
$q_0$	$\triangleright$	$(q_0, \rightarrow)$
$q_1$	$a$	$(q_0, a)$
$q_1$	$\sqcup$	$(q_0, \rightarrow)$
$q_1$	$\triangleright$	$(q_1, \rightarrow)$

Considere a fita de entrada  $\triangleright a a \square \square \square \square \square \square \square \square \square \square \dots$  um exemplo de alimentação da máquina  $M$ . Pode-se afirmar que o processamento desta fita se dará conforme a seguir.

$$\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|} \hline \Downarrow & a & a & \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \end{array} \dots, \Downarrow = q_0$$

$$\delta(q_0, \triangleright) = (q_0, \rightarrow)$$

$$\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|} \hline \triangleright & \Downarrow & a & \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \end{array} \dots, \Downarrow = q_0$$

$$\delta(q_0, a) = (q_1, \sqcup)$$

$$\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|} \hline \triangleright & \Downarrow & \square & a & \square & \square & \square & \square & \square & \square & \square \\ \hline \end{array} \dots, \Downarrow = q_1$$

$$\delta(q_1, \sqcup) = (q_0, \rightarrow)$$

$$\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|} \hline \triangleright & \square & \Downarrow & \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \end{array} \dots, \Downarrow = q_0$$

$$\delta(q_0, a) = (q_1, \sqcup)$$

$$\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|} \hline \triangleright & \square & \Downarrow & \square & \square & \square & \square & \square & \square & \square & \square \\ \hline \end{array} \dots, \Downarrow = q_1$$

$$\delta(q_1, \sqcup) = (q_0, \rightarrow)$$

$$\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|} \hline \triangleright & \square & \square & \Downarrow & \square & \square & \square & \square & \square & \square & \square \\ \hline \end{array} \dots, \Downarrow = q_0$$

$$\delta(q_0, \sqcup) = (h, \sqcup)$$

$$\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|} \hline \triangleright & \square & \square & \square & \Downarrow & \square & \square & \square & \square & \square & \square \\ \hline \end{array} \dots, \Downarrow = h$$

Observe que a máquina é iniciada no estado  $q_0$ , com a cabeça leitora  $\Downarrow$  na posição de leitura do símbolo  $\triangleright$ . Em seguida, de acordo com os símbolos lidos e a função de transição  $\delta$ , a máquina prossegue com seu processamento até que em algum momento atinge o estado  $h$ , um estado de parada.

A representação gráfica utilizada nesta descrição de processamento pode ser substituída por uma representação menos trabalhosa (*estado, situação da fita*). A representação do estágio inicial da máquina do exemplo anterior seria dada por  $(q_0, \underline{\triangleright} aa \square \square \square \square \square \square \square \square \dots)$ , onde  $\underline{\quad}$  representa a posição da cabeça leitora. Esta forma de explicitar um determinado momento da máquina será usada em detrimento da forma de representação gráfica.

Desta forma, a computação da máquina  $M$  possui seis passos é dada como se segue:

$$\begin{aligned}
(q_0, \triangleright aa \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup) &\vdash_M (q_0, \triangleright \underline{a} a \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup) \\
&\vdash_M (q_1, \triangleright \sqcup \underline{a} \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup) \\
&\vdash_M (q_0, \triangleright \sqcup \sqcup \underline{a} \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup) \\
&\vdash_M (q_1, \triangleright \sqcup \sqcup \sqcup \underline{a} \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup) \\
&\vdash_M (q_0, \triangleright \sqcup \sqcup \sqcup \sqcup \underline{a} \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup) \\
&\vdash_M (h, \triangleright \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup \sqcup)
\end{aligned}$$

Um problema de decisão é uma questão sobre um sistema formal, tal como uma Máquina de Turing, no qual se obtém uma resposta do tipo sim-ou-não, ou ainda, do tipo aceita-rejeita, ou pára-não pára. Observe que no exemplo proposto a máquina  $M$  percorre toda a fita de entrada até que finalmente atinge o estado  $h$ , definido como um estado final. Ora, se a máquina parou em um estado final, é porque ela aceitou a string a qual ela foi submetida. Neste sentido dizemos que a máquina  $m$  efetuou um *decisão*.

Entretanto, não é sempre que uma Máquina de Turing consegue realizar uma decisão. Considere uma nova máquina  $M = \langle k, \Sigma, \delta, s, F \rangle$ , onde  $k = \{q_0, q_1, q_2, h\}$ ,  $\Sigma = \{a, \sqcup, \triangleright\}$ ,  $s = q_0$ ,  $F = \{h\}$  e  $\delta$  é dada conforme a seguir:

$q$	$\sigma$	$\delta(q, \sigma)$
$q_0$	$a$	$(q_1, \leftarrow)$
$q_0$	$\sqcup$	$(q_0, \sqcup)$
$q_0$	$\triangleright$	$(q_0, \rightarrow)$
$q_1$	$a$	$(q_2, \sqcup)$
$q_1$	$\sqcup$	$(h, \sqcup)$
$q_1$	$\triangleright$	$(q_1, \rightarrow)$
$q_2$	$a$	$(q_2, a)$
$q_2$	$\sqcup$	$(q_0, \leftarrow)$
$q_2$	$\triangleright$	$(q_2, \rightarrow)$

Suponha que  $n \geq 0$ . Podemos tentar descrever o que  $M$  faz quando iniciada na configuração  $(q_0, \triangleright \sqcup a^n \underline{a})$ . Observe as iterações para alguns valores de  $n$ .

$$\begin{aligned}
& \text{Para } n = 0 \rightsquigarrow (q_0, \triangleright \sqcup \underline{a}) \vdash_M (q_1, \triangleright \sqcup \underline{a}) \\
& \vdash_M (h, \triangleright \sqcup a) \leftarrow \text{Pára} \\
& \text{Para } n = 1 \rightsquigarrow (q_0, \triangleright \sqcup a\underline{a}) \vdash_M (q_1, \triangleright \sqcup \underline{a}a) \\
& \vdash_M (q_2, \triangleright \sqcup \sqcup \underline{a}) \\
& \vdash_M (q_0, \triangleright \sqcup \sqcup a) \leftarrow \text{Loop} \\
& \text{Para } n = 2 \rightsquigarrow (q_0, \triangleright \sqcup aa\underline{a}) \vdash_M (q_1, \triangleright \sqcup a\underline{a}a) \\
& \vdash_M (q_2, \triangleright \sqcup a\underline{a}a) \\
& \vdash_M (q_0, \triangleright \sqcup \underline{a} \sqcup a) \\
& \vdash_M (q_1, \triangleright \sqcup \sqcup a \sqcup a) \\
& \vdash_M (h, \triangleright \sqcup a \sqcup a) \leftarrow \text{Pára} \\
& \text{Para } n = 3 \rightsquigarrow (q_0, \triangleright \sqcup aaa\underline{a}) \vdash_M (q_1, \triangleright \sqcup aa\underline{a}a) \\
& \vdash_M (q_2, \triangleright \sqcup aa\underline{a}a) \\
& \vdash_M (q_0, \triangleright \sqcup a\underline{a} \sqcup a) \\
& \vdash_M (q_1, \triangleright \sqcup \underline{a}a \sqcup a) \\
& \vdash_M (q_2, \triangleright \sqcup \sqcup a \sqcup a) \\
& \vdash_M (q_0, \triangleright \sqcup \sqcup a \sqcup a) \leftarrow \text{Loop}
\end{aligned}$$

Pode-se concluir que a máquina  $M$  aceita strings com  $n$  par, e que esta mesma máquina não pára, isto é, rejeita strings com  $n$  ímpar. Neste caso em que a máquina ora apresenta um comportamento de decisão, ora outro de não decisão, diz que ela faz uma *semi-decisão*. Este assunto será ainda aprofundado na seção 7.5.

As Máquinas de Turing também podem ser representadas através de diagramas de estados, com apenas algumas alterações em relação à representação dos autômatos finitos. Observe a Figura 7.1 que apresenta uma máquina segundo este tipo de representação, sendo seu estado inicial o  $q_0$ . Inicialmente, os círculos que antes continham as etiquetas descritivas dos estados, agora estão com estas marcações fora deles, como por exemplo o estado  $q_0$ . Em geral, estas marcações são descartadas, sendo utilizadas neste exemplo apenas para efeito didático. Ainda descrevendo os estados, observa-se que no interior deles existe um símbolo  $R$ , ou  $L$ . Estes símbolos

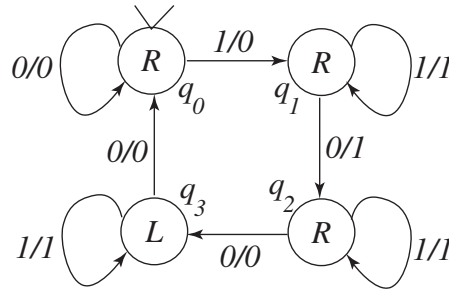


Figura 7.1: Máquina de Turing representada através de um diagrama de estados: *shiftadora*.

são os indicativos de qual movimento será realizado na cabeça leitora da máquina, isto é, direita (*R - right*) ou esquerda (*L - left*), respectivamente.

Outra questão importante são os eventos que ocorrem nas setas de transição. Nos autômatos finitos que foram estudados até o presente momento, apenas era indicado o símbolo lido associado a transição. Como as Máquinas de Turing podem não somente ler um símbolo, mas também podem escrever na fita de entrada, tem-se que esta operação mais sofisticada é representada através de uma marcação do tipo "símbolo lido, símbolo escrito". Assim 1/0 significa que o símbolo lido foi o 1, e que após esta leitura será escrito o símbolo 0 na mesma posição, alterando a fita de entrada.

Por fim, a dinâmica do processamento da Máquina de Turing se dá na seguinte ordem:

1. lê um símbolo da fita;
2. escreve um símbolo na mesma posição da leitura;
3. entra no estado indicado pela seta correspondente;
4. efetua um movimento da cabeça leitora para a direita ou para a esquerda, ditado pelo estado no qual se acabou de entrar.

Desta forma, o início do processamento da máquina descrita na Figura 7.1 pode ser explanado conforme a seguir. A máquina começa no estado inicial  $q_0$ , aguardando

saber qual símbolo será lido. Se o símbolo lido for o 0, então ela escreverá 0 na fita, desloca-se para  $q_0$  e deslocará a cabeça leitora para a próxima posição à direita. O leitor pode pensar que sair de  $q_0$  e entrar novamente em  $q_0$  significa o mesmo que permanecer em  $q_0$ . Entretanto, para auxiliar no entendimento do movimento da cabeça leitora, sugere-se esta forma de leitura do que está acontecendo. Se o símbolo lido for 1, então ela irá escrever um 0 na mesma posição, irá entrar no estado  $q_1$ , e irá efetuar um movimento da cabeça leitora para a direita.

Suponha a configuração  $(q_0, \dots \underline{0}1000 \dots)$  para a máquina da Figura 7.1. A computação desta máquina é apresentada a seguir:

$$\begin{aligned}
(q_0, \dots \underline{0}1000 \dots) &\vdash_M (q_0, \dots 0\underline{1}000 \dots) \\
&\vdash_M (q_1, \dots 000\underline{0}0 \dots) \\
&\vdash_M (q_2, \dots 001\underline{0}0 \dots) \\
&\vdash_M (q_3, \dots 00\underline{1}00 \dots) \\
&\vdash_M (q_3, \dots 00\underline{0}100 \dots) \\
&\vdash_M (q_0, \dots 00\underline{1}00 \dots) \\
&\vdash_M (q_1, \dots 000\underline{0}0 \dots) \\
&\vdash_M (q_2, \dots 0001\underline{0} \dots) \\
&\vdash_M (q_3, \dots 000\underline{1}0 \dots) \\
&\vdash_M (q_3, \dots 000\underline{0}10 \dots) \\
&\vdash_M (q_0, \dots 000\underline{1}0 \dots) \dots
\end{aligned}$$

Agora suponha a computação realizada para a um outra configuração.

$$\begin{aligned}
(q_0, \dots \underline{0}111000 \dots) &\vdash_M (q_0, \dots 0\underline{1}11000 \dots) \\
&\vdash_M (q_1, \dots 00\underline{1}1000 \dots) \\
&\vdash_M (q_1, \dots 001\underline{1}000 \dots) \\
&\vdash_M (q_1, \dots 0011\underline{0}00 \dots) \\
&\vdash_M (q_2, \dots 00111\underline{0}0 \dots) \\
&\vdash_M (q_3, \dots 00111\underline{0}0 \dots) \\
&\vdash_M (q_3, \dots 001\underline{1}100 \dots) \\
&\vdash_M (q_3, \dots 00\underline{1}1100 \dots) \\
&\vdash_M (q_3, \dots 0\underline{0}11100 \dots) \\
&\vdash_M (q_0, \dots 00\underline{1}1100 \dots) \\
&\vdash_M (q_1, \dots 000\underline{1}100 \dots) \\
&\vdash_M (q_1, \dots 0001\underline{1}00 \dots) \\
&\vdash_M (q_1, \dots 00011\underline{0}0 \dots) \\
&\vdash_M (q_2, \dots 000111\underline{0} \dots) \\
&\vdash_M (q_3, \dots 000111\underline{0} \dots) \\
&\vdash_M (q_3, \dots 0001\underline{1}10 \dots) \\
&\vdash_M (q_3, \dots 000\underline{1}110 \dots) \\
&\vdash_M (q_3, \dots 000\underline{1}110 \dots) \\
&\vdash_M (q_3, \dots 000\underline{1}110 \dots) \\
&\vdash_M (q_0, \dots 000\underline{1}110 \dots) \dots
\end{aligned}$$

Observando as duas computações que foram realizadas é possível concluir que o que está acontecendo é que a máquina pega o primeiro símbolo 1 a aparecer, e o desloca para a posição do primeiro 0, após uma cadeia de 1's. O efeito visual que ocorre é o do deslocamento à direita da seqüência de 1's, uma operação conhecida como shift à direita. Como não há estado final, a máquina permanece em loop realizando os descolamentos infinitamente.

**Exemplo 7.1** A máquina da Figura 7.2 representa uma máquina copiadora. Esta máquina copia a seqüência contígua de 1's à direita do símbolo  $A$  para a direita do símbolo  $B$ . (Sugere-se utilizar a configuração  $\dots 0\underline{A}111000B0000 \dots$  para teste).



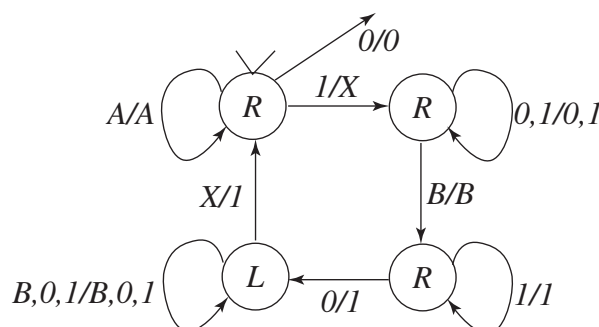


Figura 7.2: Máquina de copiadora do Exemplo 7.1.

A máquina da Figura 7.2 apresenta alguns itens de representação novos. O primeiro item, e mais importante, é a presença de um estado final. Diferente de como era realizado na representação dos autômatos finitos, nas Máquinas de Turing não é comum indicar o estado final explicitamente. Este estado é presumido a partir da seta de transição que sai do estado inicial, com uma etiqueta  $0/0$ . Outro item é uma generalização das próprias etiquetas das setas de transição, como por exemplo a  $0, 1/0, 1$ . Esta etiqueta é equivalente a duas etiquetas do tipo  $0/0$  e  $1/1$ . Para tornar o desenho menos denso, opta-se por desenhar apenas uma seta de transição, na condição de que a leitura e escrita dos símbolos são determinadas pela ordem com que eles aparecem na etiqueta.

Uma característica interessante é a possibilidade de uso de tantos símbolos quanto os que forem desejados. Observe que a fita na configuração  $\cdots 0A111000B0000\cdots$  é possível graças a um alfabeto  $\Sigma = \{0, 1, A, B\}$ . Os símbolos  $A$  e  $B$  tem a função nítida de servirem como marcadores para dar suporte ao procedimento de cópia, isto é,  $A$  indica a posição a partir da qual os símbolos devem ser copiados, e  $B$  a posição a partir da qual deve ser feita a cópia.

O projetista desta máquina aproveitou o conceito de marcação para introduzir um novo símbolo  $X$ . No movimento executado para a direita,  $X$  marca o símbolo do qual está sendo feita a cópia. No retorno da cabeça leitora, quando ele avança para a esquerda, este  $X$  indica para a máquina qual foi o símbolo copiado, e assim, ela é capaz de inferir o próximo símbolo a sofrer uma cópia. A introdução deste marcador serve como se fosse uma variável temporária na qual se armazena a posição do

elemento que está sendo copiado. Tal como uma variável temporária, ao término do processamento ela deve ser descartada pelo gerenciador de memória do compilador. Assim, se antes um símbolo 1 foi sobrescrito por um  $X$ , agora este mesmo  $X$  deve ser removido, restaurando o símbolo original. Desta forma, ao término da computação, a Máquina de Turing irá apresentar somente os símbolos pertencentes ao alfabeto  $\Sigma$ .

## 7.2 Máquinas de Turing Combinadas

A combinação de Máquinas de Turing permite que sejam criadas máquinas mais complexas a partir de máquinas mais simples. Muitas vezes é difícil conceber uma máquina que execute uma tarefa qualquer simplesmente por conta da inabilidade de quem a constrói em lidar com tantos estados e transições. Nestes casos, a estratégia de dividir para conquistar torna-se uma opção valiosa. Divide-se o problema inicial em partes menores mais simples e se realiza a construção de máquinas para cada uma destas partes individualmente, algo que a Engenharia de Software chama de análise. Logo após, realiza-se a síntese, isto é, a combinação destas partes para que juntas provejam a solução do problema inicial.

Sob esta ótica, torna-se desejável entender como combinar estas máquinas. Considere dois grupos de Máquinas de Turing com alfabeto  $\Sigma = \{a, b, \dots, \sqcup\}$ . O primeiro grupo é composto por máquinas que escrevem na fita um determinado símbolo, sobre outro já existente, e depois pára no estado  $h$ . Por exemplo, a função de transição da máquina  $M_a$  que escreve o símbolo  $a$  nestas condições (e similarmente  $M_b, \dots, M_{\sqcup}$ ) é definida como:

$q$	$\sigma$	$\delta(q, \sigma)$
$q_0$	$a$	$(h, a)$
$q_0$	$b$	$(h, a)$
$q_0$	$\vdots$	$(h, a)$
$q_0$	$\sqcup$	$(h, a)$

O segundo grupo é composto por máquinas que movem a cabeça leitora uma unidade à esquerda ( $M_L$ ) ou uma unidade à direita ( $M_R$ ) e em seguida pára no estado  $h$ . Por

# Referências Bibliográficas

- [1] HOPCROFT, J. E., *Theory os Machines and Compuatations*, chapter An n log n algorithm for minimizing states in a finite automaton, Academic Press, pp. 189–196, 1971.
- [2] MONTEIRO, A. A., PAULO, J. D. S., *Aritmética Racional*. Lisboa, Livraria Avelar Machado, 1945.
- [3] IFRAH, G., *Os Números: a história de uma grande invenção*. São Paulo, Globo S.A., 2005.
- [4] DEDEKIND, R., *Was sind und was sollen die Zahlen*, v. 3, *Gesammelte Mathematische Werke*. New York, Chelsea Publishing Company, 1969. pp. 335-391.
- [5] GÖDEL, K., *Collected Works*, v. 2, *Gesammelte Mathematische Werke*. Oxford, Oxford University Press, 1990.
- [6] ISRAEL, D., “Reflections on Gödel’s and Gandy’s Reflection on Turing’s Thesis”, *Minds and Machines*, v. 12, n. 2, pp. 181–201, 2002.
- [7] SOBRINHO, J. Z., “Aspectos da Tese de Church-Turing”, *Revista Matemática Universitária - USP*, v. 1, n. 6, pp. 1–23, 1987.
- [8] MCDERMOTT, D., “Artificial Intelligence Meets Natural Stupidity”, *SI-GART Newsletter*, v. 57, pp. 4–9, April 1976.
- [9] SETTI, M. D. O. G., *O Processo de Discretiza çã o do Raciocínio Matemático na Tradu çã o para o Raciocínio Computacional*, Report, Universidade Federal do Paraná, 2009.

- [10] GUERREIRO, G., “A Vanguarda Matemática e os Limites da Razão”, *Scientific American Brasil*, v. 5, n. 12, pp. 39–56, 2007. Coleção Gênios da Ciência.
- [11] SMULLYAN, R., *Gödel’s Incompleteness Theorems*. Oxford University Press, 1992.
- [12] GÖDEL, K., *The Undecidable*, chapter On Formally Undecidable Propositions of Principia Mathematica and Related Systems, New York, Raven Press, pp. 5–38, 1965.
- [13] WHITEHEAD, A. N., RUSSELL, B., *Principia Mathematica*. Londres, Cambridge University Press, 1913.
- [14] NAGEL, E., NEWMAN, J. R., *Gödel’s Proof*. USA, Routledge, 1989.
- [15] GÖDEL, K., “Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme - On Formally Undecidable Propositions of Principia Mathematica and Related Systems”, *Kurt Gödel: Collected Works*, v. 1, n. 1, pp. 144–195, 1986. Tradução para o inglês por Martin Hirzel. [www.research.ibm.com/people/h/hirzel/papers/canon00-goedel.pdf](http://www.research.ibm.com/people/h/hirzel/papers/canon00-goedel.pdf) [capturado em 13 de agosto de 2011].
- [16] MELO, A. C. V. D., SILVA, F. S. C. D., *Modelos Clássicos de Computação*, Coleção Schaum. São Paulo, Thomson, 2006.
- [17] KUBRUSLY, R. D. S., *Uma viagem informal ao Teorema de Gödel ou (O preço da matemática é o eterno matemático)*, Report, Universidade Federal do Rio de Janeiro, 2007. IM/UFRJ.
- [18] SMULLYAN, R., *Recursion Theory for Metamathematics*. Oxford University Press, 1993.
- [19] SMULLYAN, R., *What’s the Name of This Book*. Penguin Books, 1978.
- [20] SMULLYAN, R., *Forever Undecided: A Puzzle Guide to Gödel*. Oxford University Press, 1987.
- [21] GOLDSTEIN, R., *Incompleteness: The Proof and Paradox of Kurt Gödel*. W. W. Norton Company, Inc., 2005.

- [22] HOFSTADTER, D. R., *Gödel, Escher e Bach: an Eternal Golden Braid*. Nova Iorque, Basic Books, 1979.
- [23] Rodríguez-Consuegra, F. A. (ed.), *Kurt Gödel - Unpublished Philosophical Essays*. Berlin, Birkhäuser Verlag, 1995.
- [24] Feferman, S., *et al.* (eds.), *Kurt Gödel - Collected Works*, v. I, II, III. New York, Oxford University Press, 1986.
- [25] WANG, H., *Reflections on Kurt Gödel*. Cambridge, Massachusetts, The MIT Press, 1988.
- [26] SUPPES, P., *Axiomatic Set Theory*. New York, Dover, 1972.
- [27] LIPSCHUTZ, S., *Teoria Elementar dos Conjuntos*, Cole çã o Schaum. Sã o Paulo, McGraw-Hill do Brasil, 1990.
- [28] SUPPES, P., *Axiomatic Theory Set*. USA, Van Nostrand Company Inc., 1960.
- [29] MELLO, F. L. D., CARVALHO, R. L. D., “Knowledge Geometry”, *Journal of Information and Knowledge Management*, v. 14, pp. 1550028, 2015.
- [30] STOLL, R. R., *Set Theory and Logic*. USA, Dover Publications Inc., 1961.
- [31] MIRAGLIA, F., *Teoria dos Conjuntos: um mínimo*. Brasil, Edusp - Editora da Universidade de São Paulo, 1992.
- [32] HALMOS, P., *Teoria Ingênua dos Conjuntos*. Brasil, Edusp - Editora da Universidade de São Paulo, 1970.
- [33] GRATZER, G., *Universal Algebra*. USA, Van Nostrand Company Inc., 1968.
- [34] COHN, P. M., *Universal Algebra*. USA, Harper and Row, 1965.
- [35] GALLIER, J. H., *Logic for Computer Science*. USA, John Wiley and Sons Inc., 1987.
- [36] PREPARATTA, F. P., YEH, R. T., *Introduction to Discreate Structures for Computer Science and Engineering*. USA, Addison-Wesley Publishing Company, 1973.

- [37] SHOENFIELD, J. R., *Degrees of Unsolvability*. North-Holland Publishing Company, 1971.
- [38] BRAINERD, W. S., LANDWEBER, L. H., *Theory of Computation*. USA, John Wiley and Sons, 1974.
- [39] EILENBERG, S., ELGOT, C., *Recursiveness*. New York: Academic Press, 1970.
- [40] CARVALHO, R. L. D., OLIVEIRA, C. M. G. M. D., *Modelos de Computação e Sistemas Formais*, 11<sup>a</sup> Escola de Computação. Rio de Janeiro, Universidade Federal do Rio de Janeiro, 1998.
- [41] BRAINERD, W. S., LANDWEBER, L. H., *Theory of Computation*. John Wiley & Sons, 1974.
- [42] KLEENE, S. C., *Introduction to Metamathematics*. D. Van Nostrand Company, Inc., 1952.
- [43] Rogers Jr., H., *Theory of Recursive Functions and Effective Computability*. USA, McGraw-Hill Book Company, 1967.
- [44] DAVIS, M., *Computability and Unsolvability*. New York, Dover, 1983.
- [45] BOOLOS, G. S., JEFFREY, R. C., *Computability and Logic*. Cambridge University Press, 1974.
- [46] MARTIN DAVIS, R. S., WEYUKER, E. J., *Computability Complexity and Languages*. New York, Academic Press, 1994.
- [47] MALLOZZI, J. S., LILLO, N. J. D., *Computability with Pascal*. New Jersey, Prentice-Hall, Inc., 1984.
- [48] TURING, A. M., *The Undecidable*, chapter On Computable Numbers, with an Application to the Entscheidungsproblem, New York, Raven Press, pp. 115–151, 1965.
- [49] ELGOT, C. C., ROBINSON, A., “Random-access stored-program machines, an approach to programming languages”, *Journal of the ACM*, v. 11, pp. 365–399, 1964.

- [50] PREPARATTA, F. P., YEH, R. T., *Introduction to Discrete Structures for Computer Science and Engineering*. Addison-Wesley Publishing Company, 1973.
- [51] HENNIE, F., *Introduction to Computability*. Addison-Wesley Publishing Company, 1977.
- [52] NELSON, R. J., *Introduction to Automata*. USA, Jonh Wiley & Sons, Inc., 1968.
- [53] CARVALHO, R. L. D., *Máquinas, Programas e Algoritmos*, 2<sup>a</sup> Escola de Computação. Campinas, Universidade Estadual de Campinas, 1981.
- [54] MENDELSON, E., *Introduction to Mathematical Logic*, Cole Mathematics Series. 3 ed. The Wadsworth and Brooks, 1987.
- [55] MANIN, Y. I., *A Course in Mathematical Logic*, Graduate Texts in Mathematics 53. 1 ed. Springer-Verlag, 1977.
- [56] HOMER, S., SELMAN, A. L., *Computability and Complexity Theory*, Texts in Computer Science. 2 ed. Springer, 2011.
- [57] CUTLAND, N. J., *Computability: An introduction to recursive function theory*. Cambridge University Press, 1980.
- [58] SIPSER, M., *Introduction to The Theory of Computation*, Course Technology Series. 2 ed. Thomson, 2006.
- [59] WALTER CARNIELLI, R. L. E., *Computability: computable functions, logic and the foundations of mathematics*. Belmont, Wadsworth and Brooks, 1989.
- [60] TARSKI, A., *Logic, semantics, metamathematics*, chapter Fundamental concepts of the methodology of the deductive sciences, London, Oxford at the Clarendon Press, pp. 60–109, 1969.
- [61] ADAM YOUNG, M. Y., *Malicious Cryptography: Exposing Cryptovirology*. John Wiley and Sons Inc., 2004.

- [62] BONFANTE, G., KACZMAREK, M., MARION, J.-Y., *A Classification of Viruses through Recursion Theorems*, volume 4497 of Lecture Notes in Computer Science. 2 ed. CiE 2007, 2007.
- [63] MACHTEY, M., YOUNG, P., *An Introduction to the General Theory of Algorithms*. New York, North Holland, 1978.
- [64] ROBINSON, J. A., “A machine oriented logic based on the resolution principle”, *J. Assoc. Comput.*, v. 12, pp. 23–41, 1965.
- [65] ROBINSON, J. A., “Automatic deduction with hyper-resolution”, *Internat. J. Comput. Math*, v. 1, pp. 227–234, 1965.
- [66] GILMORE, P. C., “A proof method for quantification theory: Its justification and realization”, *IBM Journal of Research and Development*, v. 4, n. 1, pp. 28–35, 1960.
- [67] DAVIS, M., PUTNAM, H., “A computing procedure for quantification theory”, *Journal of the ACM*, v. 7, n. 3, pp. 201–215, 1960.
- [68] CHANG, C.-L., LEE, R. C.-T., *Symbolic Logic and Mechanical Theorem Proving*. Academic Press Inc, 1973.
- [69] KLEENE, S. C., *Mathematical Logic*. Wiley, 1967.
- [70] HILBERT, D., ACKERMANN, W., *Prinmciples of Mathematical Logic*. Chelsea, 1950.
- [71] MCCAWLEY, J. D., *Everything That Linguists Have Always Wanted To Know About Logic*. 2 ed. The University of Chicago Press, 1993.
- [72] SUPPES, P., *Introduction to Logic*. D. van Nostrand, 1966.
- [73] RUSSELL, B., *A Filosofia do Atomismo Lógico*, *Lógica e Conhecimento*. 1 ed. Abril Cultural, 1974. (Os Pensadores, 42).
- [74] RUSSELL, B., *Significado e Verdade*. 1 ed. Zahar, 1978.
- [75] POPPER, K. R., *A Lógica da Pesquisa Científica*. 2 ed. Cultrix, 1974.



- [76] LAKATOS, I., , MUSGRAVE, A., *A Crítica e o Desenvolvimento do Conhecimento*. 1 ed. EDUSP, Cultrix, 1979. Tradução: M. O. Caiado.
- [77] WANG, H., *From Mathematics to Philosophy*. 1 ed. Cultrix, 1974.
- [78] GREEN, C. C., *The Application of Theorem Proving to Question-Answering Systems*. Ph.D. dissertation, Stanford, June 1969. AI Project MEMO AI-96.
- [79] LOVELAND, D. W., *Automated Theorem Proving: a Logical Basis*. 1 ed. North Holland, 1978.
- [80] HUGHES, G. E., LONDEY, D. G., *The Elements of Formal Logic*. USA, Methuen and Co Ltd, 1965.
- [81] BOOK, R. V., OTTO, F., *String-Rewriting Systems*. USA, Springer-Verlag, 1993.
- [82] HOPCROFT, J. E., ULLMAN, J. D., *Introduction to Automata Theory, Language and Computation*. USA, Addison-Wesley Publishing Company, 1979.
- [83] HOPCROFT, J. E., ULLMAN, J. D., *Formal Languages and their Relation to Automata*. USA, Addison-Wesley Publishing Company, 1969.
- [84] CURRY, H. B., *Foundations of Mathematical Logic*. New York, Academic Press, 1977.
- [85] SMULLYAN, R., *Theory of Formal Systems*. USA, Princeton, 1961.
- [86] BERSTEL, J., BOASSON, L., CARTON, O., *et al.*, *Handbook of Automata: from Mathematics to Applications*, chapter Minimization of automata, European Mathematical Society, pp. 189–196, 2010.
- [87] BEAL, M. P., CROCHEMORE, M., “Minimizing incomplete automata”, *Workshop on Finite State Methods and Natural Language Processing*, , september 2008. Ispra.
- [88] VALMARI, A., LEHTINEN, P., “Efficient minimization of DFAs with partial transition”, *Proc. 25th Symp. Theoretical Aspects of Comp. Sci.*, v. 08001, pp. 645–656, 2008. S. Albers and P. Weil, editors.

- [89] PAPADONIKOLAKIS, M., BOUGANIS, C.-S., CONSTANTINIDES, G.,  
“Performance comparison of GPU and FPGA architectures for the SVM training problem”, *IEEE International Conference on FieldProgrammable Technology*, pp. 388–391, 2009.
- [90] MU, S., WANG, C., LIU, M., *et al.*, “Evaluating the potential of graphics processors for high performance embedded computing”, *Proc. IEEE Design, Automation and Test in Europe Conference and Exhibition*, pp. 709–714, 2011.
- [91] KAI HWANG, F. A. B., *Computer Architecture and Parallel Processing*. McGraw-Hill, 1984.
- [92] AGARWAL, P., KRISHNAN, S., MUSTAFA, N., *et al.*, *Algorithms - ESA 2003, Lecture Notes in Computer Science*, chapter Streaming Geometric Optimization Using Graphics Hardware, Springer Berlin-Heidelberg, pp. 115–151, 2003.
- [93] TANENBAUM, A. S., *Organização Estruturada de Computadores*. 3 ed. Prentice Hall do Brasil, 1997.
- [94] BACKUS, J., “Can Programming be Liberated from von Neumann Style? A functional style and its algebra of program”, *ACM Turing Award Lecture, Communications of the ACM*, v. 21, n. 8, pp. 613–641, 1978.
- [95] OWENS, J. D., LUEBKE, D., GOVINDARAJU, N., *et al.*, “A Survey of General-Purpose Computing on Graphics Hardware”, *Eurographics 2005, State of the Art Reports*, pp. 21–51, 2005.
- [96] GUSTAFSON, J. L., “Reevaluating Amdahl’s law”, *Communications of the ACM*, v. 5, n. 31, pp. 532, 1988.
- [97] HANDLER, W., *Parallel Processing Systems, an advanced course*, chapter Innovative computer architecture - how to increase parallelism but not complexity, Cambridge University Press, pp. 1–41, 1982.
- [98] LOBUR, J., NULL, L., *The Essentials of Computer Organization And Architecture*. Jones and Bartlett Pub, 2006.

- [99] LEWIS, H. R., PAPADIMITRIOU, C. H., *Elements of the Theory of Computation*. 2 ed. New York, Prentice-Hall, 1998.
- [100] DUNNE, P., *Computability Theory: Concepts and Applications*. Ellis Horwood, 1991.
- [101] AARONSON, S., “NP-complete Problems and Physical Reality”, *ACM SIGACT News*, , march 2005. Complexity Theory Column 46.