

2. \mathcal{R} é um conjunto de relações $r \subseteq D^n$, $n > 0$, n é a aridade de r .
3. \mathcal{F} é um conjunto de funções $f : D^n \rightarrow D$, $n \geq 0$, n é a aridade de f .

Definição 2.22 Sejam $\mathcal{E}_1 = \langle D_1, \mathcal{R}_1, \mathcal{F}_1 \rangle$ e $\mathcal{E}_2 = \langle D_2, \mathcal{R}_2, \mathcal{F}_2 \rangle$ estruturas abstratas. Diz-se que \mathcal{E}_1 é compatível com \mathcal{E}_2 se e somente se

1. Para cada $f \in \mathcal{F}_1$ de aridade n existe $f^* \in \mathcal{F}_2$ de aridade n , e
2. Para cada $r \in \mathcal{R}_1$ de aridade n existe $r^* \in \mathcal{R}_2$ de aridade n .
3. Diz-se que \mathcal{E}_1 e \mathcal{E}_2 são compatíveis se e somente se \mathcal{E}_1 é compatível com \mathcal{E}_2 e \mathcal{E}_2 é compatível com \mathcal{E}_1 .

Definição 2.23 Sejam $\mathcal{E}_1 = \langle D_1, \mathcal{R}_1, \mathcal{F}_1 \rangle$ e $\mathcal{E}_2 = \langle D_2, \mathcal{R}_2, \mathcal{F}_2 \rangle$ estruturas abstratas compatíveis e $h : D_1 \rightarrow D_2$ uma função. Diz-se que h é um homomorfismo de \mathcal{E}_1 em \mathcal{E}_2 se e somente se:

1. Para todo $f \in \mathcal{F}_1$, $f^* \in \mathcal{F}_2$ de aridade n e $a_1, a_2, \dots, a_n \in D_1$

$$h(f(a_1, a_2, \dots, a_n)) = f^*(h(a_1), h(a_2), \dots, h(a_n))$$

2. Para todo $r \in \mathcal{R}_1$, $r^* \in \mathcal{R}_2$ de aridade n e $a_1, a_2, \dots, a_n \in D_1$

$$\langle a_1, a_2, \dots, a_n \rangle \in r \Rightarrow \langle h(a_1), h(a_2), \dots, h(a_n) \rangle \in r^*$$

Se h é uma bijeção diz-se que h é um isomorfismo de \mathcal{E}_1 em \mathcal{E}_2 . Note que se existe um isomorfismo h de \mathcal{E}_1 em \mathcal{E}_2 , h é uma bijeção, portanto h^{-1} é um isomorfismo de \mathcal{E}_2 em \mathcal{E}_1 . Diz-se que \mathcal{E}_1 e \mathcal{E}_2 são estruturas abstratas isomorfas.

Estruturas abstratas $\langle D, \mathcal{R}, \mathcal{F} \rangle$, tal que $\mathcal{R} = \{=\}$ são chamadas de estruturas algébricas ou simplesmente álgebras.

2.6 Indução Finita

As propriedades principais do conjunto dos números naturais são:

1. Os números naturais podem ser gerados a partir do número natural 0 via a operação de sucessor.
2. Quando uma propriedade numérica ocorre para um número natural, e também ocorre para o próximo número natural da geração, então a propriedade acontecerá para todos os números naturais subseqüentes.

A segunda propriedade, chamada de princípio da indução finita, é tão geral que merece uma consideração especial.

Princípio da indução finita *Seja P uma propriedade de números naturais. Se 0 possui a propriedade P , e ainda, quando n possui a propriedade a mesma P juntamente com $n + 1$, então tem-se que todo natural tem a propriedade P .*

O princípio da indução é usado para demonstrar asserções P sobre números naturais, e o procedimento de demonstração tem os seguintes passos:

- (a) Base da indução (BI): mostrar que 0 satisfaz a asserção P .
- (b) Hipótese de indução (HI): supor que o número natural k satisfaz a asserção P , e demonstrar que:
- (c) Passo de indução (PI): $k + 1$ satisfaz a asserção P
- (d) Conclusão da indução (CI): de (a), (b) e (c) concluir que todo natural n satisfaz a asserção P .

Exemplo 2.18 Prove que $0 + 1 + 2 + 3 + \dots + k = \frac{k \cdot (k+1)}{2}$.

BI Se $k = 0$ temos $0 = \frac{0 \cdot (0+1)}{2}$

HI Suponha válido para $k = n$, ou seja $0 + 1 + \dots + n = \frac{n \cdot (n+1)}{2}$

PI Para $k = n + 1$, temos

$$\begin{aligned}
 0 + 1 + 2 + \dots + n + (n + 1) &= \frac{n \cdot (n+1)}{2} + (n + 1) \quad \text{por HI} \\
 &= \frac{n \cdot (n+1)}{2} + \frac{2 \cdot (n+1)}{2} \\
 &= \frac{(n+1) \cdot (n+2)}{2} \\
 &= \frac{(n+1) \cdot ((n+1)+1)}{2}
 \end{aligned}$$

CI A propriedade é válida.

Para que o princípio da indução finita possa ser utilizado na demonstração de uma propriedade de um conjunto devemos ter que este é **bem-ordenado**, como os \mathbb{N} . Um conjunto A é bem-ordenado se existe uma relação de ordem total \preceq para A e todo subconjunto não-vazio X de A possui um único elemento minimal.

Em geral, podemos aplicar o princípio da indução a conjuntos **bem fundados**, um conceito mais abrangente que bem-ordenado. Um conjunto A é bem-fundado se existe uma relação de ordem parcial \preceq para A e todo subconjunto não-vazio X de A possui um elemento minimal. Neste caso pode-se ter um número infinito de elementos minimais. Esta propriedade é fundamental para que seja possível utilizar indução em um conjunto pois caso contrário não se poderia verificar a base de indução, que são os elementos minimais (no caso do conjunto \mathbb{N} , o 0). Para os conjuntos bem-fundados tem-se o seguinte princípio geral da indução, ou indução completa.

Princípio geral da indução *Seja $< A, \preceq >$ uma ordem com A bem-fundado e seja P uma propriedade dos elementos de A . Se todo $l \preceq k$, $l, k \in A$, tem a propriedade P , e k tem a propriedade P então todo $x \in A$ tem a propriedade P .*

Indução é muito utilizada para definir objetos, conjuntos, relações e funções. Isto acontece quando temos uma expressão geral que define tais indivíduos como membros de uma classe ou família. Observe o exemplo a seguir.

Exemplo 2.19 Seja $A = \{\square, \diamond\}$, C o conjunto de *concatenações* dos elementos de A e $F = \{f\}$ onde:

$$\begin{aligned} f(\square x) &= \square x \diamond \\ f(\diamond x) &= \diamond x \square \end{aligned}$$

assim tem-se a seguinte definição de um conjunto chamado O :

- (a) $A = \{\square, \diamond\} \subseteq O$;
- (b) Se $x \in O$ então $f(x) \in O$;
- (c) Os únicos elementos de O são os objetos satisfazendo os itens (a) e (b) acima.

É fácil verificar que $\square\diamond, \diamond\square$ são elementos de O , porém $\square\square$ não é um elemento de O .

No Exemplo 2.19, o conjunto O é indutivo em A . O item a. é a cláusula básica da definição, o item b. é a cláusula de indução e c. a cláusula de fechamento. A é o conjunto básico ou inicial e f é a função geradora. Diz-se também que O é definido por indução a partir de A por f .

Observe no próximo exemplo, a definição indutiva dos números naturais a partir da função sucessor.

Exemplo 2.20 O conjunto dos números naturais \mathbb{N} é uma classe indutiva. O conjunto A é o conjunto $\{0\}$, e a função é a operação de sucessor, que soma 1 a um número natural:

- (a) 0 é um número natural;
- (b) Se a é um número natural então o sucessor de a é um número natural.
- (c) Os únicos números naturais são os objetos satisfazendo os itens (a) e (b) acima

Os exemplos 2.19 e 2.20 obedecem ao esquema geral de definições indutivas, apresentado a seguir.

Definição 2.24

Conjunto Indutivo Sejam D e $A \subset D$ conjuntos, e F um conjunto de funções $f : D^k \rightarrow D$, $k > 0$. Diz-se que um conjunto B é indutivo em A , se e somente:

1. $A \subseteq B$;
2. Se $a_1, a_2, \dots, a_k \in B$ então $f(a_1, a_2, \dots, a_k) \in B$ para toda $f \in F$;

Fecho Indutivo Diz-se que B é definido por indução se B é a interseção de todos os conjuntos indutivos em A , também chamado “fecho indutivo” de A sob F .

Observe que o fecho indutivo é o menor conjunto indutivo em A . Alternativamente, pode-se ter uma definição mais construtiva do fecho indutivo de um conjunto, como se segue.

Definição 2.25 Sejam D e $A \subset D$ conjuntos, e F um conjunto de funções $f : D^k \rightarrow D$, $k > 0$. Dizemos que um conjunto B é definido por indução em A , se e somente existe uma seqüência B_0, B_1, \dots tal que:

1. $B_0 = A$
2. $B_{i+1} = B_i \cup \{f(a_1, a_2, \dots, a_k) | f \in F, a_1, a_2, \dots, a_k \in B_i\}$
3. $B = \bigcup_{i \geq 0} B_i$

Fica para o leitor a demonstração de que as duas definições acima se equivalem. Frequentemente serão definidos conjuntos de objetos possuindo uma estrutura formal, tal como o conjunto do exemplo 2.19. Pretende-se demonstrar propriedades destes conjuntos.

Exemplo 2.21 Sejam os conjuntos A, C e O como definidos no exemplo 2.19. Então, todo elemento de O é $\underbrace{\square \diamond \dots \diamond}_n$ ou $\diamond \underbrace{\square \dots \square}_n$, $n \in \mathbb{N}$.

- (a) Base de Indução (BI): é válido para o conjunto inicial $A = \{\square, \diamond\}$: basta fazer $n = 0$
- (b) Hipótese de Indução (HI): supondo a_1 satisfaz a propriedade para algum n
- (c) Passo de Indução (PI): então:

1. $f(\underbrace{\square \diamond \dots \diamond}_n) = \underbrace{\square \diamond \dots \diamond \diamond}_{n+1}$; ou
2. $f(\diamond \underbrace{\square \dots \square}_n) = \diamond \underbrace{\square \dots \square \square}_{n+1}$

satisfazem a propriedade para $n + 1$.

- (d) Conclusão de Indução (CI): (a), (b) e (c) satisfazem a asserção.

2.7 Alfabetos e Linguagens

Os dados no computador são organizados através de seqüências de bits, representados pelos símbolos 0 e 1. Sob esta ótica, entende-se como desejável estudar

a matemática de cadeias de símbolos. Desta forma, define-se como alfabeto (Σ) um conjunto, em geral finito, de símbolos (σ) chamados de letras. Entende-se por símbolo todo sinal gráfico satisfazendo os seguintes critérios:

1. As letras devem possuir uma estrutura espacial que facilite sua reprodução e reconhecimento. Por exemplo: $\square, \triangle, |, *$. Como contra-exemplo o leitor pode imaginar figuras complicadas como rubricas pessoais, tais como $\mathfrak{b}\mathfrak{h}$.
2. As letras devem possuir uma estrutura que impossibilite decomposições horizontais. Assim, “ $||$ ” não seria uma escolha apropriada, pois é composta horizontalmente de dois sinais iguais (“ $|$ ” e “ $|$ ”); no entanto, “ $-$ ” e “ $==$ ” seriam escolhas adequadas - apesar de poderem ser decompostas verticalmente.
3. Para a construção de alguns sistemas, existe a necessidade de um suprimento infinito de letras, assim deve-se exigir que elas possam ser produzidas de modo uniforme.

$\Sigma_1 = \{*, |\}$ e $\Sigma_2 = \{\square, \triangle, \odot\}$ satisfazem os critérios 1., 2. e 3. descritos acima.

Exemplo 2.22 O alfabeto romano é dado por $\{a, b, \dots, z\}$, o alfabeto binário é composto por $\{0, 1\}$, e alfabeto hexadecimal por $\{0, 1, \dots, 9, A, B, \dots, F\}$

Expressões ou cadeias são seqüências de letras justapostas horizontalmente, tendo seus limites claramente identificados por inter espaço separador - com mesma função que o espaço em branco na escrita convencional. Assim, uma string em um alfabeto Σ é uma seqüência finita de símbolos deste alfabeto. O comprimento de uma string w , denotado por $|w|$, é a contagem de símbolos pertencentes a string.

Exemplo 2.23 $|101| = 3$ e $|\text{inconstitucionalissimamente}| = 27$

Uma string que não possua nenhum símbolo, isto é, de comprimento 0, é chamada de string vazia, e é denotada por Λ . Assim, seja Σ um alfabeto, uma cadeia (expressão, string) em Σ é:

1. Λ , é uma cadeia nula e seu comprimento $|\Lambda| = 0$, ou

2. uma letra $\sigma \in \Sigma$ e seu comprimento $|\sigma| = 1$.
3. Se x é uma cadeia em Σ e $\sigma \in \Sigma$ uma letra, então $x\sigma$ é uma cadeia em Σ . Se o comprimento $|x| = n$ então o comprimento de $|x\sigma| = n + 1$.

Deste modo, $\square\Delta$ e $\square\odot\square\Delta$ são expressões no alfabeto Σ_2 . Além disto, o conjunto de todas as strings, incluindo Λ , sobre o alfabeto Σ é denotado por Σ^* . Assim, o conjunto Σ^* para $\Sigma = \{a, b\}$ é dado por $\Sigma^* = \{\Lambda, a, b, aa, bb, ab, ba, aaa, \dots\}$

A operação de *concatenação* que se resume a justapor um símbolo a uma cadeia. Além disto, esta operação é associativa. Por exemplo, a concatenação de $\square\Delta$ com \square é a cadeia $\square\Delta\square$. Podemos generalizar esta operação para a concatenação de duas cadeias; assim o comprimento da cadeia resultante é a soma dos comprimentos das cadeias concatenadas. Por exemplo, $\square\Delta$ concatenada com $\square\odot\square\Delta$ é $\square\Delta\square\odot\square\Delta$.

Palavras são expressões que respeitam determinados critérios explícitos para sua formação. Desta forma pode-se explicitar o seguinte critério de formação para as palavras em Σ_1 :

Critério 1: As únicas palavras são as expressões onde não apareçam mais que duas ocorrências sucessivas de “*” e não menos que duas de “|” em seqüência.

Utilizando-se este critério, conclui-se (informalmente) que, $**||*$ é uma palavra, e que $**|*$ não é, pois possui menos que duas ocorrências sucessivas de “|”.

Seja a string w . Um símbolo $\sigma \in \Sigma$ na posição $j \in [1, |w|]$ é chamado de ocorrência de σ , e é denotado por $w(j) = \sigma$. A palavra *casa*, por exemplo, possui as seguintes ocorrências: $w(1) = c$, $w(2) = a$, $w(3) = s$ e $w(4) = a$. É importante ressaltar que as ocorrências $w(2)$ e $w(4)$ estão associadas a um mesmo símbolo a , porém, ainda assim são ocorrências distintas.

O inverso de uma string w , denotado por w^R , é a string escrita de trás para frente. Por exemplo, para a string $w = casa$, tem-se que seu inverso é dado por $w^R = asac$. Neste sentido a definição 2.26 apresenta a formalização deste conceito.

Definição 2.26 O inverso w^R de uma string w é definido como:

1. Se $|w| = 0$ então $w^R = w = \Lambda$;
2. Se $|w| = n + 1 > 0$, $w = ua$, $a \in \Sigma$ então $w^R = au^R$;

Algumas strings recebem nomes particulares de acordo com suas propriedades. As strings de letras nas quais $w^R = w$ são chamadas de palíndromes, como por exemplo, “reviver”, “osso”, “arara”. A cultura popular também consagrou algumas expressões como palíndromes, ainda que não detanhem o rigor matemático aqui apresentado, por conta da presença de espaços em branco. Entre estas, podemos citar “luz azul”, “socorram-me subi no ônibus em Marrocos” e “seco de raiva coloco no colo caviar e doces”.

As strings de números, por sua vez, nas quais $w^R = w$ são chamadas de capicua, como por exemplo, 10022001 (que poderia ser a data 10Fev2001), ou ainda o número 9 no alfabeto binário (1001).

Exemplo 2.24 Mostre que dados duas strings quaisquer x e w , $(xw)^R = w^R x^R$, como por exemplo $(casa)^R = (sa)^R (ca)^R = asac$.

Demonstração por Indução

1. Base de Indução (BI):

$$|x| = 0 \rightarrow x = \Lambda$$

$$(xw)^R = (\Lambda w)^R = w^R = w^R \Lambda = w^R \Lambda^R = w^R x^R$$

2. Hipótese de Indução (HI): $(xw)^R = w^R x^R$ para $|x| = n - 1$

3. Passo de Indução (PI):

$$\text{Seja } y = ax, |y| = n, |x| = n - 1 \text{ e } |a| = 1$$

$$(yw)^R = ((ax)w)^R = (a(xw))^R = (xw)^R a = w^R x^R a = w^R (ax)^R = w^R y^R$$

4. Conclusão de Indução (CI): (1), (2) e (3) demonstram a asserção.

Definição 2.27 [Linguagem] Seja Σ um alfabeto. Então:

1. Define-se o conjunto Σ^* da seguinte maneira: seja $\Sigma^0 = \{\Lambda\}$ e $\Sigma^{i+1} = \Sigma^i \cup \{x\sigma \mid x \in \Sigma^i \text{ e } \sigma \in \Sigma\}$, então $\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^i$. Ou seja, Σ^* é o conjunto de todas as cadeias em Σ .

2. Diz-se que \mathcal{L} é uma **linguagem** em Σ se e somente se $\mathcal{L} \subseteq \Sigma^*$ (em geral, uma linguagem é um conjunto definido por uma propriedade).

Exemplo 2.25 Dado o alfabeto $\Sigma = \{\square, \triangle, \odot\}$

$$\mathcal{L} = \{\square, \square\triangle, \square\odot, \triangle\}$$

é uma linguagem em Σ .

2.8 Objetos Finitos e Espaços

Um objeto é finito se sua especificação requer apenas uma quantidade finita de informação. Um número natural é um objeto finito que pode ser especificado pela sua representação arábica. Nem todo número real é um objeto finito. Por exemplo, existem números reais transcendentos, cuja especificação poderia ser dada apenas exibindo uma série infinita de termos⁵. Uma n -tupla de objetos finitos é também um objeto finito, desde que pode ser especificada pela enumeração finita de suas n componentes. Uma classe finita de objetos finitos é também um objeto finito, mas uma classe infinita de objetos em geral não é um objeto finito. Um **espaço** é qualquer conjunto X de objetos finitos tal que, dado um objeto finito qualquer, sempre é possível verificar efetivamente sua pertinência a X . Por exemplo:

1. A classe de números naturais;
2. Se A e B são espaços então $A \times B$;
3. A classe dos números naturais divisível por 5.

Note que a classe de funções $f : \mathbb{N} \rightarrow \mathbb{N}$ não é um espaço pois seus objetos não são em geral finitos. Um dos objetivos deste livro é estudar que classe de funções possui representações finitárias, sendo, portanto objetos finitos.

⁵É claro que os transcendentos conhecidos têm uma especificação aparentemente finita, por exemplo π , como sendo a razão entre o dobro do comprimento de qualquer circunferência para o seu raio, mas estamos aqui aceitando como um fato que trata-se de um caso particular e não uma regra.

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 [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^a 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^a 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.