

Se $x = 2$, então

$$\begin{aligned} P(x) \rightarrow Q(f(x), a) &= P(2) \rightarrow Q(f(2), a) \\ &= P(2) \rightarrow Q(1, 1) \\ &= 1 \rightarrow 1 = 1 \end{aligned}$$

Uma vez que $P(x) \rightarrow Q(f(x), a)$ é 1 para todos os elementos x no domínio D , a fórmula $G = 1$ sob a interpretação I .

4.4 Forma Normal Prenex

Em lógica proposicional foram introduzidos os conceitos de formas normais conjuntivas e disjuntivas. Em lógica de primeira ordem também existe uma forma normal, neste caso conhecida como *forma normal prenex*. A razão principal para se considerar fórmulas escritas em forma normal prenex deve-se às tentativas de simplificar o processo de prova, conforme será visto mais adiante.

Definição 4.11 [Forma Normal Prenex] A fórmula F em lógica de primeira ordem é dita esta em forma normal prenex se e somente a fórmula F encontra-se segundo o padrão $(Q_1x_1) \dots (Q_nx_n)(M)$, onde (Q_ix_i) pode ser tanto $(\forall x_i)$ como $(\exists x_i)$, e M uma fórmula sem quantificador algum. $(Q_1x_1) \dots (Q_nx_n)$ são chamados de prefixo e M de matriz da fórmula F .

Deste modo, na Fórmula Normal Prenex, o escopo dos quantificadores deve ser a fórmula inteira. Assim, a fórmula $(\forall x)(\forall y)(P(x, y) \wedge Q(y))$ e a fórmula $(\forall x)(\forall y)(\exists z)(Q(x, y) \rightarrow R(z))$ encontram-se na forma normal prenex. As regras de equivalência para normalização previstas na Tabela 4.5 de lógica proposicional continuam válidas para a lógica de primeira ordem. Entretanto, para a lógica de primeira ordem também existem algumas outras regras de equivalência válidas, tal como apresentado na Tabela 4.9, cujas as demonstrações são simples e deixadas para o leitor. Na tabela, considere que G é uma fórmula que não contém a variável x , e ainda, Q pode ser tanto o quantificador \forall , como o \exists .

Neste momento, é importante observar que o quantificador universal \forall e o

Tabela 4.9: Regras de equivalência para normalização prenex.

$$\begin{aligned}
r.1a) \quad & (Qx)F[x] \vee G \equiv (Qx)(F[x] \vee G) \\
r.1b) \quad & (Qx)F[x] \wedge G \equiv (Qx)(F[x] \wedge G) \\
r.2a) \quad & \neg((\forall x)F[x]) \equiv (\exists x)(\neg F[x]) \\
r.2b) \quad & \neg((\exists x)F[x]) \equiv (\forall x)(\neg F[x]) \\
r.3a) \quad & (\forall x)F[x] \wedge (\forall x)H[x] \equiv (\forall x)(F[x] \wedge H[x]) \\
r.3b) \quad & (\exists x)F[x] \vee (\exists x)H[x] \equiv (\exists x)(F[x] \vee H[x])
\end{aligned}$$

quantificador existencial \exists não são distributivos sobre os conectivos \vee e \wedge , respectivamente. Isto é,

$$\begin{aligned}
(\forall x)F[x] \vee (\forall x)H[x] &\neq (\forall x)(F[x] \vee H[x]) \\
(\exists x)F[x] \wedge (\exists x)H[x] &\neq (\exists x)(F[x] \wedge H[x])
\end{aligned}$$

Para casos como estes é preciso usar de alguns artifícios. Uma vez que a variável ligada na fórmula pode ser considerada como uma variável qualquer (ordinária), toda variável x pode ser renomeada para z , e a fórmula $(\forall x)H[x]$ é alterada para $(\forall z)H[z]$. Suponha que foi escolhida uma variável z que não aparece em $F[x]$. Assim tem-se,

$$\begin{aligned}
(\forall x)F[x] \vee (\forall x)H[x] &\equiv (\forall x)F[x] \vee (\forall z)H[z] \\
&\equiv (\forall x)(\forall z)(F[x] \vee H[z])
\end{aligned}$$

Similarmente tem-se,

$$\begin{aligned}
(\exists x)F[x] \wedge (\exists x)H[x] &\equiv (\exists x)F[x] \wedge (\exists z)H[z] \\
&\equiv (\exists x)(\exists z)(F[x] \wedge H[z])
\end{aligned}$$

De modo geral, o processo pra transformar fórmulas quaisquer para uma representação na forma normal prenex segue um processo sistemático, tal como

descrito no Algoritmo 4.1

Data: Fórmula da lógica de 1ª ordem desnormalizada

Result: Fórmula da lógica de 1ª ordem na forma normal prenex

while *existem conectivos \rightarrow e \leftrightarrow* **do**

Substituir: $F \leftrightarrow G \equiv (F \rightarrow G) \wedge (G \rightarrow F)$;

Substituir: $F \rightarrow G \equiv \neg F \vee G$;

end

while \neg *não está o mais próximo possível dos átomos* **do**

Substituir: $\neg(\neg F) \equiv F$;

Substituir: $\neg(F \vee G) \equiv \neg F \wedge \neg G$;

Substituir: $\neg(F \wedge G) \equiv \neg F \vee \neg G$;

Substituir: $\neg((\forall x)F[x]) \equiv (\exists x)(\neg F[x])$;

Substituir: $\neg((\exists x)F[x]) \equiv (\forall x)(\neg F[x])$;

end

Renomear as variáveis ligadas se necessário;

while *os quantificadores não estiverem todos na esquerda da fórmula* **do**

Substituir: $(Qx)F[x] \vee G \equiv (Qx)(F[x] \vee G)$;

Substituir: $(Qx)F[x] \wedge G \equiv (Qx)(F[x] \wedge G)$;

Substituir: $(\forall x)F[x] \wedge (\forall x)H[x] \equiv (\forall x)(F[x] \wedge H[x])$;

Substituir: $(\exists x)F[x] \vee (\exists x)H[x] \equiv (\exists x)(F[x] \vee H[x])$;

Substituir: $(Q_1x)F[x] \vee (Q_2x)H[x] \equiv (Q_1x)(Q_2z)(F[x] \vee H[z])$;

Substituir: $(Q_3x)F[x] \wedge (Q_4x)H[x] \equiv (Q_3x)(Q_4z)(F[x] \wedge H[z])$;

end

Algoritmo 4.1: Transformação de fórmulas para a forma normal prenex

Exemplo 4.7 Transforme a fórmula $(\forall x)P(x) \rightarrow (\exists x)Q(x)$ para forma normal prenex.

$$\begin{aligned} G &= (\forall x)P(x) \rightarrow (\exists x)Q(x) \\ &\equiv \neg((\forall x)P(x)) \vee (\exists x)Q(x) \\ &\equiv (\exists x)(\neg P(x)) \vee (\exists x)Q(x) \\ &\equiv (\exists x)(\neg P(x) \vee Q(x)) \end{aligned}$$

Exemplo 4.8 Obtenha a forma normal prenex para a fórmula $(\forall x)(\forall y)((\exists z)P(x, z) \wedge$

$$P(y, z) \rightarrow (\exists u)Q(x, y, u).$$

$$\begin{aligned} G &= (\forall x)(\forall y)((\exists z)P(x, z) \wedge P(y, z)) \rightarrow (\exists u)Q(x, y, u) \\ G &\equiv (\forall x)(\forall y)(\neg((\exists z)P(x, z) \wedge P(y, z)) \vee (\exists u)Q(x, y, u)) \\ &\equiv (\forall x)(\forall y)((\forall z)(\neg P(x, z) \vee \neg P(y, z)) \vee (\exists u)Q(x, y, u)) \\ &\equiv (\forall x)(\forall y)(\forall z)(\exists u)(\neg P(x, z) \vee \neg P(y, z) \vee Q(x, y, u)) \end{aligned}$$

4.5 Forma de Normal de Skolen

Nas seções anteriores foi estudado como transformar uma fórmula na forma normal prenex, e também descrito como transformar a matriz em forma normal conjuntiva. Nesta seção será verificado como eliminar os quantificadores existenciais empregando a forma normal de Skolen¹. Observe que se G está na Forma Normal de Skolen se ela é oriunda de uma Prenex H , cujos quantificadores existenciais (\exists) foram retirados por Skolem. É importante observar que $G \not\equiv H$, mas H é insatisfatível se e somente se G também for, e esta é a razão principal para estudá-la.

Seja uma formula F já na forma normal prenex $(Q_1x_1) \dots (Q_nx_n)M$, onde M é uma forma normal conjuntiva. Suponha que Q_r seja um quantificador existencial no prefixo $(Q_1x_1) \dots (Q_nx_n)$, $1 \leq r \leq n$. Se nenhum quantificador universal estiver a esquerda de Q_r , escolhe-se uma nova constante c diferente das outras constantes de M , substitui-se por c todas as ocorrências de x_r em M e apaga-se Q_rx_r do prefixo. Se Q_{s1}, \dots, Q_{sm} são quantificadores universais que antecedem Q_r , $1 \leq s_1 < s_2 < \dots < s_m < r$, escolhe-se uma função de aridade m diferente dos outros símbolos funcionais, substitui-se todos os x_r em M por $f(x_{s1}, x_{s2}, \dots, x_{sm})$ e apaga-se (Q_rx_r) do prefixo. Quando este processo alcança todos os quantificadores existenciais do prefixo, diz-se que a fórmula F encontra-se na forma de Skolen. As funções e as constantes utilizadas na substituição dos quantificadores existenciais são chamadas de funções de skolenização.

Exemplo 4.9 Obtenha a forma de Skolen para a fórmula:

$$(\exists x)(\forall y)(\forall z)(\exists u)(\forall v)(\exists w)P(x, y, z, u, v, w)$$

¹Thoralf Albert Skolem(23 de maio de 1887 - 23 de março de 1963), matemático norueguês.

Nesta fórmula, $(\exists x)$ não é precedido por qualquer quantificador universal, $(\exists u)$ é precedido por $(\forall y)$ e $(\forall z)$, enquanto que $(\exists w)$ é precedido por $(\forall y)$, $(\forall z)$ e $(\forall v)$. Deste modo, substitui-se a variável existencial x pela constante a , u pela função de aridade dois $f(y, z)$ e w pela função de aridade três $g(y, z, v)$. Assim, obtém-se a seguinte fórmula: $(\forall y)(\forall z)(\forall v)P(a, y, z, f(y, z), v, g(y, z, v))$.

Exemplo 4.10 Obtenha a forma de Skolen para a fórmula:

$$(\forall x)(\exists y)(\exists z)((\neg P(x, y) \wedge Q(x, z)) \vee R(x, y, z))$$

Primeiro a matriz é transformada em forma normal conjuntiva:

$$(\forall x)(\exists y)(\exists z)((\neg P(x, y) \vee R(x, y, z)) \wedge (Q(x, z) \vee R(x, y, z)))$$

Enfim, uma vez que $(\exists y)$ e $(\exists z)$ são precedidos por $(\forall x)$, as variáveis existenciais y e z são substituídas, respectivamente pelas funções $f(x)$ e $g(x)$. Deste modo, obtém-se a seguinte forma de Skolen:

$$(\forall x)((\neg P(x, f(x)) \vee R(x, f(x), g(x))) \wedge (Q(x, g(x)) \vee R(x, f(x), g(x))))$$

Seja, por exemplo, a fórmula $(\forall x)(\exists y)\text{é_parente-de}(y, x)$, cujo entendimento é que toda pessoa x tem ao menos um parente y . Se simplesmente trocarmos para $(\forall x)\text{é_parente-de}(\text{João}, x)$ acabaria-se por fazer uma interpretação incorreta uma vez que esta fórmula indica que João é parente de todas as pessoas x . A interpretação correta é que a variável x deveria ser substituída por um "parente genérico" do domínio de y , sem ser uma variável. Desta forma, é razoável fazer $y = f(x)$, pois y depende de x . Logo a substituição correta é $(\forall x)\text{é_parente-de}(f(x), x)$.

Definição 4.12 [Cláusula] Uma cláusula é uma disjunção de literais.

As disjunções $\neg P(x, f(x)) \vee R(x, f(x), g(x))$ e $Q(x, g(x)) \vee R(x, f(x), g(x))$ na forma de skolen do Exemplo 4.10 são cláusulas. Um conjunto S de cláusulas é o conjunto de todas as cláusulas, onde cada variável em S é considerada como sendo quantificada por um quantificador universal. Deste modo, as formas de Skolen podem ser escritas como um conjunto de cláusulas. Por exemplo, a forma de Skolen do Exemplo

4.10 pode ser representada pelo conjunto $\{\neg P(x, f(x)) \vee R(x, f(x), g(x)), Q(x, g(x)) \vee R(x, f(x), g(x))\}$. O conjunto S de cláusulas será um ponto chave no procedimento para prova automática de teoremas.

4.6 Teorema de Herbrand

Até o presente momento, foi apresentado um ferramental de suporte para ser empregado na prova de teoremas. Nesta seção serão considerados os procedimentos desta prova. A tarefa de se encontrar um procedimento para averiguar a validade (consistência) de uma fórmula foi uma tarefa estudada no passado. Primeiramente foi tentado por Leibniz, e em seguida por Peano quase duzentos anos depois, próximo a virada para o século XX. Na década de 1920 foram dados os primeiros passos efetivos neste sentido por Herbrand, mas foi só em 1936, através de Church e Turing que isso se mostrou ser uma tarefa impossível. De forma independente, tanto Church como Turing mostraram que não existe um procedimento genérico para checar a validade de fórmulas em lógica de primeira ordem. Entretanto, para o caso específico no qual as fórmulas são de fato válidas, existem procedimentos que podem chegar a esta conclusão. Para o caso em que as fórmulas não são válidas, estes procedimentos nunca se encerram. Sob a ótica de Church e Turing, está é, na melhor das hipóteses, a situação limite que se pode atingir com estes procedimentos de prova.

Uma abordagem importante para a prova automática de teoremas foi dada por Herbrand² em 1931. Por definição, uma fórmula é válida se ela assumir valores 1 para todas as interpretações. Herbrand propôs um algoritmo para encontrar uma interpretação que refuta uma determinada fórmula. Todavia, se uma fórmula realmente é válida, tal interpretação não deverá existir, e seu algoritmo deverá abortar após um número finito de tentativas. Seu método era impraticável de se aplicar até a invenção do computador digital. Só após o artigo de Robinson [64, 65] em 1965, junto com o desenvolvimento do princípio da resolução, foi possível o desenvolvi-

²Jacques Herbrand (1908-1931) foi um matemático francês, cujo estudo principal foi a teoria de demonstração e as funções recursivas gerais intitulado "On the consistency of arithmetic". Em uma escalada dos Alpes franceses com dois amigos, caiu nas montanhas de granito do Massif des Écrins e morreu. "On the consistency of arithmetic" foi publicado postumamente.

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 Discrete 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.