Sintaxe e semântica formais de "Máquinas de Post" como máquinas de estados finitos*

Ana Paula Lüdtke Ferreira[1], **Nicholas Cortez Moreira**[2], **Lourenço Nataniel Pinheiro Portella**[2]

[1]Programa de Pós-graduação em Computação Aplicada
Universidade Federal do Pampa (UNIPAMPA) – Campus Bagé
anaferreira@unipampa.edu.br

[2]Bacharelado em Engenharia de Computação
Universidade Federal do Pampa (UNIPAMPA) – Campus Bagé
{nicholasmoreira.aluno,lourencoportella.aluno}@unipampa.edu.br

Resumo. *O estudo da equivalência de modelos de Computação requer que a sintaxe e a semântica dos modelos sejam formalmente definidas. Neste trabalho investigamos como a "Máquina de Post" aparece nos referenciais bibliográficos usados em um conjunto de IES com cursos na área de Computação e propomos definições sintáticas e semânticas rigorosas para o modelo. O resultado é um modelo mais similar ao de Turing, facilitando a construção e o entendimento de provas de equivalência.*

Abstract. *The equivalence proof of Computing models requires their formal syntax and semantics to be defined. In this work, we investigate how the "Post machine" appears in the bibliographic references used in a set of HEIs with courses in the area of Computing, and we propose a rigorous syntactic and semantic definition for it. The result is a model more similar to Turing's, facilitating the construction and understanding of equivalence proofs.*

1. Introdução

As Diretrizes Curriculares Nacionais para cursos da área de Computação, instituídas pela Resolução CNE/CES nº 5/2016, exigem que os egressos "dominem os fundamentos teóricos da área de Computação e como eles influenciam a prática profissional" (Art. 4º, inciso IV do §1º). Usualmente, fundamentos da Computação referem-se ao estudo da complexidade de algoritmos e de problemas, modelos de computação e computabilidade.

O suporte bibliográfico a modelos de computação define a abrangência e a profundidade dos temas trabalhados. Diferentes modelos são apresentados para estabelecer sua equivalência ao modelo de Turing. Funções μ-recursivas de Gödel/Kleene e o Cálculo λ de Church possuem uma sintaxe (funções e termos algébricos) distante do conceito de máquina de estados finitos. Outros modelos que seguem a semântica de máquina de estados são muitas vezes apresentados com uma sintaxe que não possibilita ao estudante enxergar com clareza a equivalência entre os modelos, muito menos construir uma prova formal de equivalência.

*Trabalho concluído, com autores discentes de graduação.

O chamado modelo de Post, conforme referenciado em livros didáticos de Teoria da Computação ([Martin 2010, pag. 413], [Diverio and Menezes 2011, pag. 183]) usa uma fila como estrutura auxiliar de memória. A apresentação mais completa do modelo usa fluxogramas e uma definição semiformal para sua sintaxe [Diverio and Menezes 2011], mesmo sendo sua operação dada por um sistema de transição de estados. Provas de equivalência necessitam que a semântica da operação das máquinas seja formalmente definida, do contrário só é possível fazer argumentações informais. Ainda que estudantes de Computação não sejam Matemáticos em formação, a Matemática está na gênese da Computação e provas de teoremas fazem parte do estudo de seus fundamentos.

O modelo original de Post [Post 1936] é bastante similar ao modelo de Turing e não usa filas em sua construção. Fizemos uma revisão bibliográfica tentando descobrir onde e quando uma máquina que usa uma fila como estrutura auxiliar de memória passou a ser chamada de "máquina de Post", sem sucesso. Entramos em contato com o *National Institute of Standards and Technology* (NIST), que mantém um dicionário de Algoritmos e Estruturas de Dados e que apresentava a definição de Máquina de Post como "similar ao modelo de Turing, mas com uma fila em vez de uma fita infinita", referenciando o trabalho original de Post. A entrada a respeito da estrutura (`https://xlinux.nist.gov/dads/HTML/postMachine.html`) foi alterada em decorrência da busca infrutífera que daí decorreu. Uma observação de que "o trabalho original apresenta um modelo similar ao de Turing, com infinitas caixas em qualquer direção" foi inserida após a referência ao trabalho de Post. Essa é a razão para que o termo apareça entre aspas no título do trabalho. Usaremos o termo sem aspas a partir daqui, visto que ele aparece em outros referenciais e também porque não encontramos a primeira referência a um autômato com fila como estrutura auxiliar.

Este artigo visa definir uma Máquina de Post como uma máquina de estados finitos, similar à definição usada para Máquinas de Turing. O método fez uso de uma revisão de textos didáticos usados em instituições com cursos de Computação bem avaliados no país, buscando entender como esse modelo de computação é mencionado nas obras. A falta de definições sintáticas e semânticas encontradas motivou este trabalho, visando tornar a noção de equivalência entre modelos mais clara para os estudantes, uma vez que a similaridade da descrição entre os dois modelos torna as provas de equivalência mais simples. O uso da formulação aqui apresentada pode ser usada por professores do tema, para trabalhar conceitos de modelos de computação em suas disciplinas.

O restante do texto está estruturado como se segue: a Seção 2 apresenta o conjunto de material e métodos do trabalho; a Seção 3 define formalmente uma Máquina de Post como uma máquina de estados finitos, incluindo sua semântica de execução; a Seção 4 apresenta as conclusões do trabalho.

2. Referenciais bibliográficos sobre Computabilidade

A revisão da literatura buscou os livros utilizados pelas 10 Instituições de Ensino tradicionalmente melhor colocadas nos rankings disponíveis de cursos de Engenharia e Ciência da Computação, como o Ranking Universitário da Folha [Folha de São Paulo 2019] e com os dados de IGC (índice geral de cursos) e CPC (conceito preliminar de curso) do INEP. Nomeadamente: Unicamp, UFMG, UFRGS, UFRJ, UFPE, USP, PUC-Rio, UFSCAR,

UFPR e UFSC. A escolha foi feita porque não seria possível analisar todos os PPC de cursos na área no Brasil e acredita-se que essas instituições sejam representativas de cursos de qualidade na área de Teoria da Computação.

O foco da revisão foi verificar como os diferentes referenciais abordam os temas relacionados à Tese de Church, e quais são os modelos de computação suportados na bibliografia. Entende-se que a abordagem utilizada em cada obra é dada pelos autores, em acordo aos seus objetivos. O nosso objetivo foi verificar se e como os modelos que usam uma fila como estrutura auxiliar são apresentados e como as noções de equivalência de modelos são desenvolvidas.

O método de trabalho incluiu a listagem das bibliografias básica e complementar das disciplinas que abordam os tópicos sobre Teoria da Computação (modelos de computação e computabilidade) nas Universidades selecionadas. A Tabela 1 apresenta os resultados.

Tabela 1. Referenciais bibliográficos em disciplinas de Teoria da Computação

Obra	**Bibliografia básica**	**Bibliografia complementar**
[Sipser 2006]	Unicamp, UFMG, UFPE, UFSCAR, UFPR E UFSC	UFRGS, UFRJ e USP
[Lewis and Papadimitriou 2004]	Unicamp, PUC-Rio e UFSCAR	UFRGS, UFPE e UFPR
[Vieira 2006]	Unicamp	UFMG e UFPR
[Hopcroft et al. 2001]	USP, Unicamp, UFSCAR, UFPR e UFSC	UFRGS, UFPE, USP e PUC-Rio
[Diverio and Menezes 2011]	UFRGS e PUC-Rio	–
[S. C. Coutinho 2007]	UFRJ	–
[Sudkamp 2005]	UFPR	–
[Harrison 1978]	Unicamp	–
[Greenlaw and Hoover 1998]	Unicamp	–
[Floyd and Beigel 1994]	Unicamp	–
[Minsky 1967]	Unicamp	–
[Bird 1976]	UFRGS	–
[Menezes 2002]	–	UFSCAR, UFPR e UFSC
[Martin 2010]	–	UFPR
[Moll et al. 1988]	–	USP
[Boolos et al. 2007]	–	PUC-Rio
[Machtey and Young 1978]	–	PUC-Rio
[Kozen 1997]	–	UFPE

Após o levantamento dos referenciais bibliográficos usados, selecionou-se as cinco obras mais frequentes, para análise do conteúdo. Os resultados são apresentados a seguir e sumarizados na Tabela 2.

[Sipser 2006] apresenta todas as classes de linguagens da Hierarquia de Chomsky, mas com um foco maior em complexidade, quando comparado aos outros textos. Máquinas de Turing são formalmente apresentadas, mas somente como forma de definir classes de complexidade, nos capítulos seguintes. Extensões de máquinas de Turing,

Referência	Máquina de Turing	Modelo de Post	Provas de equivalência
[Sipser 2006]	Sintaxe e semântica formais	Exercício	–
[Lewis and Papadimitriou 2004]	Sintaxe e semântica formais	Referências	Estrutura das provas
[Vieira 2006]	Sintaxe e semântica formais	Citado no texto	Estrutura das provas
[Hopcroft et al. 2001]	Sintaxe e semântica formais	Referências	Estrutura das provas
[Diverio and Menezes 2011]	Sintaxe formal, semântica informal	Seção dedicada	Argumentação informal

Tabela 2. Conteúdos das obras detalhadas

autômatos de um número k de pilhas e máquinas com uma fila com estrutura auxiliar (chamada de autômato de fila) aparecem nos exercícios do capítulo. Não são dadas provas de equivalência de modelos de computação, visto que esse não é o foco do livro.

[Lewis and Papadimitriou 2004] aborda as classes de linguagens da Hierarquia de Chomsky, com um capítulo final que trata sobre complexidade. Máquinas de Turing têm sua sintaxe e semântica formais definidas e a operação de composição de máquinas é usada para a construção de máquinas mais complexas. Outros modelos de computação aparecem no texto: autômatos de duas pilhas como um exercício para mostrar equivalência de máquinas e máquinas de registradores como uma extensão de Máquinas de Turing denominada *random access Turing machines*. As provas de equivalência têm sua estrutura apresentada com detalhe, mas nenhuma prova completa é desenvolvida. O trabalho de Post é referenciado no livro, mas somente como informação ao leitor interessado.

[Vieira 2006] detalha os mecanismos geradores e reconhecedores da Hierarquia de Chomsky. O modelo de Post é citado junto aos demais modelos computacionais contemporâneos (i.e., décadas de 30 e 40 do Século XX): máquina de Turing, λ-cálculo e funções μ-recursivas, sem detalhamento. Máquinas de Turing e suas extensões aparecem no texto com um tratamento formal, tanto da sintaxe quanto da semântica. Não são apresentados outros modelos de computação. As provas de equivalência são delineadas, com argumentações sobre como uma máquina pode simular a outra. Nenhuma prova completa é apresentada no texto e a razão disso é discutida no capítulo sobre Matemática Discreta, afirmando-se que provas devem ser apresentadas para o público a que se destinam, e que provas podem ser apresentadas informalmente para "não especialistas".

[Hopcroft et al. 2001] incorpora aplicações práticas da teoria de linguagem formais, como XML e construção de analisadores léxicos e sintáticos. Os exemplos e motivações no livro são próximos à experiência de programação de alunos de graduação. Máquinas de Turing têm um capítulo dedicado. Construções conhecidas de linguagens de programação são apresentadas como máquinas de Turing. Autômatos de duas pilhas aparecem, junto com a indicação de como provar de que eles têm o mesmo poder computacional de máquinas de Turing. Máquinas de registradores são denominadas *counter machines* e, novamente, a prova de sua equivalência com linguagens recursivamente enumeráveis é dada com uma argumentação semiformal. O trabalho de Post aparece nas

referências, mas não é discutido no corpo do texto.

[Diverio and Menezes 2011] foca seu primeiro capítulo em diferentes formas de equivalência de programas/máquinas, bastante similar à abordagem de [Bird 1976]. As provas de equivalência são indicadas e não construídas de forma completa. A sintaxe formal de máquinas de Turing é definida, mas sua semântica é dada de maneira informal. Este texto parece ser o único que usa o termo "máquina universal" para se referir a modelos de computação com o mesmo poder computacional de máquinas de Turing. A máquina de Post é apresentada em uma seção dedicada, com sintaxe semiformal e semântica informal. Não há provas de equivalência entre o modelo de Post e os demais modelos.

3. Modelo de Post como uma máquina de estados finitos

O matemático Emil Leon Post, no seu artigo *Finite Combinatory Processes Formulation 1* [Post 1936], define um modelo computacional determinístico que, como ele se refere, seria uma *working hypothesis* do Teorema da Incompletude de Gödel, demonstrado em 1931. O modelo proposto por Post nesse trabalho difere substancialmente de sua apresentação atual como uma máquina que usa uma fila como estrutura auxiliar, conforme discutido na Seção 1. Assim, o termo "máquina de Post" se refere a uma máquina de estados que opera sobre uma estrutura do tipo fila (FIFO).

A definição formal para a máquina de Post aparece na Definição 1 e sua semântica e noções usuais relacionadas a funções computadas, decisão e semidecisão de linguagens aparecem a seguir.

Definição 1 (Máquina de Post) *Uma* Máquina de Post *é uma tupla* $P = (K, \Gamma, \delta, s, H, \#)$ *em que*

- K *é um conjunto finito de estados,*
- Γ *é o alfabeto da fila,*
- $\delta : K \times \Gamma \to K \times \Gamma^*$ *é a função de transição,*
- $s \in K$ *é o estado inicial,*
- $H \subseteq K$ *é o conjunto dos estados de parada,*
- $\# \in \Gamma$ *e o símbolo de marcação de final de entrada.*

A Definição 1 apresenta uma estrutura bastante similar às definições usuais de máquina de Turing. A forma da função de transição elimina a necessidade de categorizar estados entre leitura e escrita na fila: cada transição lê (destrutivamente) e escreve alguma coisa da fita, podendo não escrever nada ou qualquer outra quantidade de símbolos. O símbolo $\#$ marca o final da palavra de entrada e é necessário para que o modelo seja Turing-completo.

Para maior clareza na representação gráfica da máquina, cada transição $\delta(p, \sigma) = (q, w)$ será representada como aparece na Figura 1(a). Uma transição da máquina de Turing $M = (K, \Sigma, \delta, s, H, \sqcup)$[1], $\delta(p, \sigma) = (q, \lambda, m)$, com $p, q \in K$, $\sigma, \lambda \in \Sigma$ e $m \in \{\leftarrow, \rightarrow, \downarrow\}$ é representada como na Figura 1(b)[2].

A noção de configuração da máquina de Post (Definição 2) contém o estado corrente e o conteúdo da fila, que são suficientes para representar toda a computação ocorrida até qualquer dado momento.

[1] K é um conjunto finito de estados, Σ é o alfabeto da fita, $\delta : K \times \Sigma \to K \times \Sigma \times \{\leftarrow, \rightarrow, \downarrow\}$ é a função de transição, $s \in K$ é o estado inicial, $H \subseteq K$ é o conjunto de estados de parada e $\sqcup \in \Sigma$ é o símbolo

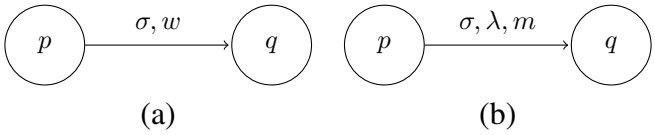

Figura 1. Formato das transições nas máquinas de Post (a) e Turing (b)

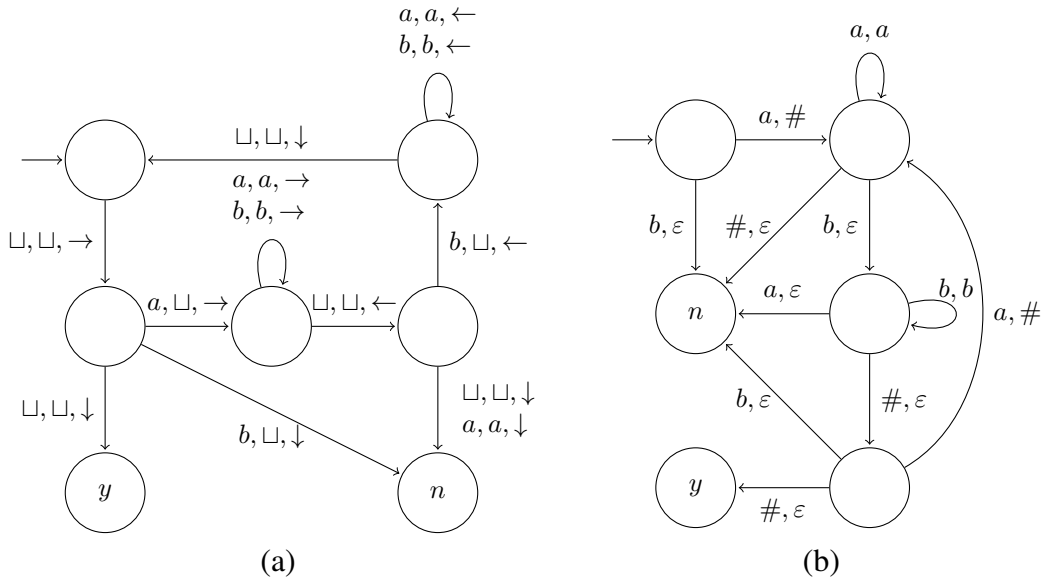

Figura 2. Máquinas de Turing (a) e de Post (b) que decidem a linguagem $L = \{a^n b^n \mid n \geq 0\}$

Definição 2 (Configuração) *Seja* $P = (K, \Gamma, \delta, s, H, \#)$ *uma Máquina de Post. Uma* configuração *de* P *é qualquer elemento de* $K \times \Gamma^*$.

As relações de transição entre configurações, em um passo e em zero ou mais passos (Definição 3), têm uma construção bastante direta e mostram as modificações na fila a cada transição aplicada.

Definição 3 (Relações $\vdash, \vdash^*$) *Seja* $P = (K, \Gamma, \delta, s, H, \#)$ *uma Máquina de Post e sejam* $(p, u), (q, v) \in K \times \Gamma^*$ *duas configurações de* P. *Diz-que* P transita em um passo *de* (p, u) *para* (q, v), *denotado por* $(p, u) \vdash_P (q, v)$ *se e somente se* $u = \sigma x$, $v = xw$ *e existe uma transição* $\delta(p, \sigma) = (q, w)$ *em* P, *para* $\sigma \in \Gamma$ *e* $u, x, v, w \in \Gamma^*$.

A relação $\vdash_P^*$ *é o fecho transitivo e reflexivo da relação* $\vdash_P$.

Uma computação de uma máquina P é uma sequência de configurações. Se a computação for finita, existe um valor $n \in \mathbb{N}$ tal que $c_1 \vdash_P c_2 \vdash_P \ldots \vdash_P c_n$ é verdadeira para uma sequência de configurações $c_1, \ldots, c_n$. Se a computação for infinita, esse valor não existe. Note-se que sem a definição formal de configuração e de transição em

branco.

[2]A definição de máquina de Turing que usamos usa uma fita infinita para os dois lados e permite a leitura, escrita e movimentação em uma só transição. A movimentação pode ser para esquerda (←), para a direita (→) ou manter o cabeçote de leitura no mesmo lugar (↓). Essa definição é equivalente a todas as outras encontradas na literatura, mas gera máquinas mais compactas.

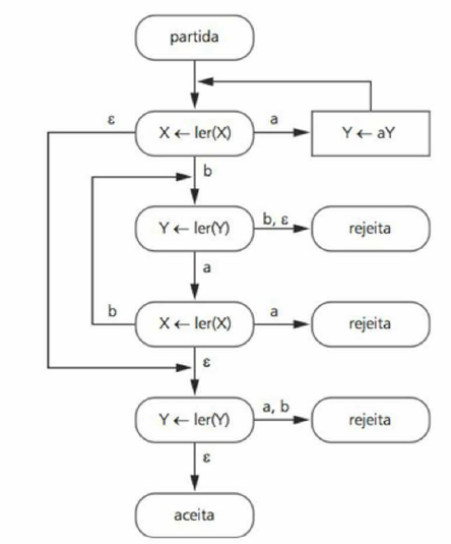

Figura 3. Máquina de Post que decide a linguagem $\mathbf{L = \{a^n b^n \mid n \geq 0\}}$ [Diverio and Menezes 2011]

um passo não é possível definir formalmente o que é uma computação, inviabilizando quaisquer provas que envolvam equivalências entre modelos de computação.

A noção de função computada pela máquina pode ser formalmente expressa como se segue:

Definição 4 (Função computada por uma Máquina de Post) *Seja $P = (K, \Gamma, \delta, s, H, \#)$ uma Máquina de Post e sejam $A, B \subseteq \Gamma$. Diz-se que P* computa uma função $f : A \to B$ *se e somente se para todo $w \in A$ existe uma computação $(s, w) \vdash_P^* (h, f(w))$, para algum $h \in H$ e $f(w) \in B$.*

As noções de decisão e semidecisão de linguagens podem ser definidas de forma similar.

Definição 5 (Linguagem decidida por Máquina de Post) *Seja $P = (K, \Gamma, \delta, s, \{y, n\}, \#)$ uma Máquina de Post e seja $\Sigma \subseteq \Gamma$. Diz-se que P* decide uma linguagem $L \subseteq \Sigma^*$ *se e somente se para todo $w \in L$ existe uma computação $(s, w) \vdash_P^* (y, u)$, para algum $u \in \Sigma^*$, e para todo $w \notin L$ existe uma computação $(s, w) \vdash_P^* (n, u)$, para algum $u \in \Sigma^*$.*

Definição 6 (Linguagem semidecidida por Máquina de Post) *Seja $P = (K, \Gamma, \delta, s, H, \#)$ uma Máquina de Post e seja $\Sigma \subseteq \Gamma$. Diz-se que P* semidecide uma linguagem $L \subseteq \Sigma^*$ *se e somente se $w \in L$ se e somente se existe uma computação $(s, w) \vdash_P^* (h, u)$, para algum $h \in H$ e $u \in \Sigma^*$.*

A Figura 2 apresenta as máquinas de Turing (a) e Post (b) que decidem a linguagem $L = \{a^n b^n \mid n \geq 0\}$, na notação proposta neste trabalho. A similaridade visual com o modelo de Turing, bem como com os modelos de autômatos finitos e autômatos de pilha, usualmente estudados, é bem maior do que com a notação de [Diverio and Menezes 2011] (Figura 3). As definições formais da semântica da máquina permitem que traduções formais entre os modelos sejam implementados e provas de correção produzidas.

4. Conclusão

Máquinas de Turing são formalmente definidas nos referencias bibliográficos a respeito de Teoria da Comutação, permitindo a construção de provas de equivalência entre modelos, sejam modificações da própria máquina de Turing ou outros modelos de computação. As provas de equivalências, contudo, exigem definições formais da semântica da execução e das linguagens decididas ou semidecididas e das funções computadas pelas máquinas. Acreditamos que os estudantes podem se beneficiar mais das competências adquiridas ao longo de seus estudos se forem expostos a mais provas formais.

A sobrecarga cognitiva causada por modelos muito diferentes de computação pode ser uma das fontes de dificuldades encontradas pelos estudantes. Notações mais próximas dos modelos já estudados, para a partir delas se construírem modelos provadamente equivalentes (como máquinas de registradores e linguagens de programação de baixo e alto nível) nos parece um caminho de formação mais consistente.

Embora seja capaz de ilustrar conceitos como computabilidade e reconhecimento de linguagens, a máquina de Post não parece ser utilizada nas instituições de ensino brasileiras, a julgar pela bibliografia analisada. A simplicidade do modelo gera uma prova de equivalência com Turing mais direta, constituindo-se uma ferramenta importante para a compreensão dos tópicos complexos vistos pelos estudantes nas disciplinas de teoria da computação. Esperamos que este trabalho possa ser útil para docentes que ministram cursos sobre Teoria da Computação.

Referências

Bird, R. (1976). *Programs and machines*. Wiley series in computing. John Wiley & Sons, Chichester, England.

Boolos, G. S., Burgess, J. P., and Jeffrey, R. C. (2007). *Computability and Logic*. Cambridge University Press, Cambridge, England, 5 edition.

Diverio, T. A. and Menezes, P. B. (2011). *Teoria da computação: máquinas universais e computabilidade*. Bookman, 3th edition.

Floyd, R. W. and Beigel, R. (1994). *The language of machines*. W.H. Freeman, New York, NY.

Folha de São Paulo (2019). Ranking Universitário Folha 2019. https://ruf.folha.uol.com.br/2019/ranking-de-cursos/computacao/.

Greenlaw, R. and Hoover, H. J. (1998). *Modern Computability Theory*. Morgan Kaufmann, Oxford, England.

Harrison, M. A. (1978). *Introduction to formal language theory*. Addison-Wesley series in computer science. Addison Wesley Longman Publishing, New York, NY.

Hopcroft, J. E., Ullman, J. D., and Motwani, R. (2001). *Introdução à Teoria de Autômatos, Linguagens e Computação*. Campus, 2th edition.

Kozen, D. (1997). *Automata and Computability*. Undergraduate Texts in Computer Science. Springer, New York, NY, 1 edition.

Lewis, H. R. and Papadimitriou, C. H. (2004). *Elementos da Teoria da Computação*. Bookman, 2th edition.

Machtey, M. and Young, P. (1978). *An introduction to the general theory of algorithms*. North Holland.

Martin, J. C. (2010). *Introduction to Languages and the Theory of Computation*. Mc Graw Hill, 4th edition.

Menezes, P. B. (2002). *Linguagens Formais e Autômatos: Volume 3 da Série Livros Didáticos Informática UFRGS*. Bookman.

Minsky, M. L. (1967). *Computation: Finite and Infinite Machines*. Automatic Computation S. Prentice Hall, Old Tappan, NJ.

Moll, R. N., Arbib, M. A., and Kfoury, A. J. (1988). *An introduction to formal language theory*. AKM Series in Theoretical Computer Science. Springer, New York, NY, 1988 edition.

Post, E. L. (1936). Finite combinatory process: formulation 1. *The Journal of Symbolic Logic*, 1(3):103–105.

S. C. Coutinho (2007). *Autômatos e Linguagens Formais*. Universidade Federal do Rio de Janeiro.

Sipser, M. (2006). *Introduction to the Theory of Computation*. Thompson Course Technology, Boston, MA, 2 edition.

Sudkamp, T. A. (2005). *Languages and machines*. Pearson, Upper Saddle River, NJ, 3 edition.

Vieira, N. J. (2006). *Introdução aos fundamentos da computação: linguagens e máquinas*. Thomson.