Uma definição simplificada para o estudo das propriedades dos Autômatos Finitos Adaptativos

(18 de Janeiro de 2009)

Diego Queiroz

***Resumo*— Autômatos Finitos Adaptativos são máquinas capazes de reconhecer quaisquer linguagens que possam ser reconhecidas por quaisquer outros modelos computacionais conhecidos. No entanto, embora seu funcionamento seja intuitivo e de fácil entendimento, poucos materiais têm explorado uma maneira de representação simplificada dessa classe de autômatos para a utilização em aplicações menos especializadas sobre o assunto. Deste modo, este artigo tem o intuito de propor uma representação alternativa para os mesmos de modo a facilitar a sua descrição e seu estudo em projetos futuros.**

Palavras chave*— Autômatos *(Automata),* Máquina de Turing *(Turing Machine),* Teoria da Computação *(Computation Theory).

I. Introdução

Os Autômatos Adaptativos [2] são uma classe de autômatos com um poder computacional superior a maioria das outras classes de autômatos. Sua característica peculiar reside na capacidade deles poderem modificar a sua própria estrutura durante sua execução. Essas alterações são realizadas através de ações adaptativas associadas a algumas das transições do autômato que permitem que sejam incluídas ou removidas transições entre seus estados.

A partir destes, é possível definir uma especialização através dos Autômatos Finitos Adaptativos (AFA). Estes autômatos possuem as mesmas características dos Autômatos Adaptativos com a imposição de algumas restrições que os tornam muito semelhantes aos Autômatos Finitos. De fato, os AFA podem ser definidos como uma generalização dos Autômatos Finitos, uma vez que, removendo as suas ações adaptativas, temos um Autômato Finito [1], no entanto, seu poder computacional é muito maior: devido a sua característica de adaptatividade, pode ser provado que os Autômatos Finitos Adaptativos são equivalentes, em poder computacional, a Máquinas de Turing [4]. A figura 1 representa a relação entre estas classes de autômatos.

Essa característica, embora dê grandes poderes aos Autômatos Adaptativos, traz também grandes consequências: todas as condições que cercam as Máquinas de Turing, especialmente sobre incomputabilidade, também cercam os Autômatos Adaptativos (caso contrário, bastaria que se transformasse o problema na forma de Máquina de Turing em outro na forma de Autômato Adaptativo para resolver diversos problemas que até o presente momento não possuem solução).

No entanto, as definições sobre Autômatos Adaptativos em uso são demasiadamente complexas para aplicações simples, portanto, nota-se uma carência de uma definição que contemple grande parte das características dessa classe de autômatos de uma forma prática e intuitiva.

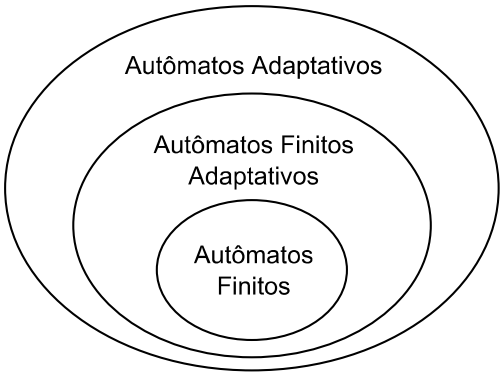

Fig. 1. Relação entre as diferentes classes de autômatos apresentados.

Com base nisso, este trabalho tem o intuito de sugerir uma formulação alternativa e detalhada para a representação de Autômatos Adaptativos com base na formulação original em [3], [11] e no modelo utilizado em [1], com o intuito de futuramente poder estudar as propriedades dos mesmos.

II. Autômatos Finitos Adaptativos

A definição mais simples de um autômato é através da sua representação como uma máquina de estados finitos, ou seja, uma máquina que, partindo de um estado inicial e consumindo uma cadeia de entrada, transita entre seus estados de acordo com um conjunto de regras de transição pré-determinadas.

Os autômatos podem possuir um ou mais estados denominados estados finais. Portanto, se após consumir uma cadeia de entrada, um autômato para sobre um destes estados, diz-se que a cadeia é reconhecida pelo autômato. Ao conjunto de cadeias que a máquina reconhece dá-se o nome de linguagem, de modo que cada tipo de máquina reconhece apenas uma classe de linguagens específica e nada mais. Essas classes são geralmente definidas com base na hierarquia de Chomsky que determina, para cada classe de linguagem, um conjunto de regras gramaticais que, quando aplicadas, geram apenas cadeias daquela classe.

Existem diversos tipos de representações diferentes de autômatos, desde as mais simples, contendo apenas os elementos básicos de uma máquina e uma quantidade finita de

Este trabalho foi financiado pelo Conselho Nacional de Desenvolvimento Científico (CNPq).

estados (Autômatos Finitos), como máquinas mais complexas que dispõem de memória auxiliar (Autômatos de Pilha) ou que incorporam funções diferenciadas [6], [7].

Embora haja equivalência entre diversos tipos de máquinas, cada classe de linguagem necessita de um tipo especializado de máquina para fazer o seu reconhecimento. No entanto, como as linguagens são definidas com base em uma hierarquia, se um tipo de máquina reconhece uma classe de linguagem também reconhece todas as classes de linguagens que estão abaixo dela hierarquicamente.

Para o reconhecimento das linguagens do topo da hierarquia, isto é, as mais gerais, chamadas de linguagens recursivamente enumeráveis, foi proposta por Turing uma "máquina de computação" [5], [6], [7] que atualmente é conhecida como Máquina de Turing. Além da simples interpretação das máquinas como reconhecedores de linguagens, acredita-se que as máquinas de Turing são versões formais de algoritmos e que, portanto, elas exprimem o limite de tudo o que pode ser computado, isto é, se algo pode ser computado, então pode ser representado por uma máquina de Turing e vice-versa.

Os Autômatos Finitos Adaptativos são máquinas de estados capazes de computar a mesma classe de linguagens que as Máquinas de Turing [4]. Sua formação é intuitivamente compreendida como um autômato finito com a capacidade de auto-modificação. Todas as suas modificações são realizadas através de ações adaptativas anexadas às transições, dessa forma, à medida que um símbolo é lido o autômato tem a habilidade de se "adaptar" para reconhecer a cadeia.

Portanto, se explorarmos esse conceito ao considerar que o autômato antes da execução da ação adaptativa é diferente do autômato após a execução da mesma [1], chegamos à conclusão que a capacidade de auto-modificação do autômato finito adaptativo faz com que este sempre possa se transformar em outro autômato à medida que transita através de estados. Desse modo, a capacidade computacional do autômato finito adaptativo reside no fato dele reunir todos os autômatos finitos em um único autômato, dispondo, indiretamente, de uma memória finita, porém, ilimitada, assim como a Máquina de Turing, estando essa, no entanto, incorporada à sua estrutura.

O modelo formal dos autômatos adaptativos pode ser representado de diversas formas. Em [2], [3], [9] os autômatos adaptativos são definidos com base em dispositivos adaptativos dirigidos por regras cujo dispositivo subjacente é um autômatos; em [1], [11] são utilizadas formalizações mais aplicadas e semelhantes às definições originais de autômatos; e, em [12], os mesmos são estudados na forma de gramáticas.

Neste artigo será apresentada uma notação simplificada dos Autômatos Finitos Adaptativos Determinísticos com o objetivo de generalizar suas funções e permitir que suas propriedades possam ser analisadas separadamente.

III. Autômatos Finitos Adaptativos Determinísticos

Diversos trabalhos destacam a importância de se incorporar o determinismo a autômatos [9], e diversos outros trabalhos já realizaram estudos comparativos sobre a complexidade de autômatos determinísticos e suas contrapartes não-determinísticas [13], [14].

Para uma máquina de estados finitos, definimos o determinismo como um requisito que faz com que haja apenas uma regra de transição para cada configuração possível.

Portanto, um Autômato Finito Adaptativo Determinístico ($AFAD$) é uma 5-tupla definida por $(Q_0, \Sigma, \delta_0, q_0, F)$ tal que:

- Q_0 é o conjunto inicial de estados;
- Σ é o alfabeto de entrada;
- δ_0 é a função de transição adaptativa inicial ($\delta_t: Q_t \times \Sigma \rightarrow Q_\infty \times \Omega^\star$) [15];
- q_0 é o estado inicial ($q_0 \in Q_0$);
- F é o conjunto de estados finais ($F \subseteq Q_0$).

com

$$\Omega = Q_\infty \times \Sigma \times Q_\infty \times \Omega^\star$$

$$Q_\infty = \bigcup_{t=0}^{\infty} Q_t \cup \{d\}$$

Em cada instante t, o $AFAD$ assume uma configuração $c_t = (Q_t, q, w, \delta_t, \bar{q})$ que determina o seu conjunto de estados no instante t, seu estado atual, o conteúdo da cadeia a ser processada, uma função de transição adaptativa e uma referência para o estado recém referenciado pela função adaptativa. Assim, sendo w uma cadeia de entrada sobre o alfabeto Σ ($w \in \Sigma^\star$), ao ser iniciado ($t = 0$) a configuração do $AFAD$ é $c_0 = (Q_0, q_0, w, \delta_0, q_0)$.

Sob essas condições, a função δ_t fornece as regras para a mudança de estado e fornece um mecanismo para a execução das ações adaptativas. Estando o $AFAD$ no estado $q \in Q_t$ e possuindo o símbolo σ na cabeça da fita de entrada, a função $\delta_t(q, \sigma) = (q_n, \psi)$ define o novo estado do $AFAD$ e a sequência de ações adaptativas a serem executadas.

Definimos "d" como sendo um estado implícito que garante o determinismo do autômato. Embora este seja geralmente omitido para facilitar a notação, este estado está presente em grande parte das definições não-determinísticas de autômatos. De um modo geral, ele é um estado especial não final ($d \notin F$) tal que a função de transição mapeia ele mesmo para todos os símbolos do alfabeto, isto é, $\delta(d, \sigma) = d$, $\forall \sigma \in \Sigma$. Desse modo, este pode ser considerado um estado terminal de *erro* do autômato finito adaptativo determinístico, pois faz este esgotar a cadeia de entrada e parar em um estado que não corresponde aos critérios para que a cadeia seja reconhecida.

Diferente da definição original dos autômatos finitos adaptativos, onde são previstas as ações de inserção e remoção [9], esta definição prevê para o autômato finito adaptativo determinístico apenas a ação de *alteração*. Essa ação adaptativa tem influência sobre a configuração posterior da máquina e faz com o autômato, ao consumir um símbolo em um estado especificado, transite para um novo estado,

[15] $A^\star$ representa o fecho reflexivo e transitivo sobre o conjunto A.

também especificado. Dessa forma, a ação adaptativa $\omega \in \Omega$ é representada pela sequência (q_i, σ, q_j, ψ) tal que:

- q_i é o estado de origem da transição que será alterada ($q_i \in (Q_\infty \cup \{\alpha,\star\}) - \{d\}$);
- σ é o rótulo da transição que receberá o novo destino;
- q_j é o novo estado de destino da transição que será alterada ($q_j \in Q_\infty \cup \{\alpha,\star\}$);
- ψ é a sequência de ações adaptativas a ser alocada a transição ($\psi \in \Omega^\star$).

Assim sendo, a ação adaptativa $\omega = (q_i, \sigma, q_j, \psi_\omega)$ fará com que a transição que sai do estado q_i, consumindo o símbolo σ da cadeia de entrada, aponte para o estado q_j associando a sequência de ações adaptativas ψ_ω. Isto é, se δ_{t-1} é a função de transição contida na configuração c_{t-1} que corresponde ao estado da máquina antes da execução da ação adaptativa e δ_t é a função de transição adaptativa contida na configuração c_t que corresponde ao estado da máquina após a execução da ação adaptativa, ω fará com que $\delta_t(q_i, \sigma) = (q_j, \psi_\omega)$ e $\delta_t(q, x) = \delta_{t-1}(q, x), \forall q \in Q_t$ e $\forall x \in \Sigma$.

Nessa formulação, dois símbolos especiais foram introduzidos: "α" e "$\star$". O símbolo "α" representa o estado atual do autômato, isto é, o estado que o autômato atingiu antes de executar a sequência de ações adaptativas. Já o símbolo "$\star$" denota uma referência a um novo estado (se $q_j = \star$) ou ao último estado criado (se $q_i = \star$). Note que $\{\alpha,\star\} \notin Q_\infty$.

Com isso, a relação binária $\vdash_{AFAD}$ entre duas configurações c_j e c_k é verdadeira se for possível passar de uma configuração para outra em apenas um movimento. Portanto, um movimento $(Q_j, q_j, w_j, \delta_j, \bar{q}_j) \vdash (Q_k, q_k, w_k, \delta_k, \bar{q}_k)$ ocorre se, e somente se, todas as condições descritas nas expressões (1), (2), (3), (4) e (5) são verdadeiras simultaneamente:

$$w_j = \sigma \circ w_k \tag{1}$$

$$\delta_j(q_j, \sigma) = (q_k, \psi) \tag{2}$$

$$\begin{gathered}(\bar{q}_j = \bar{q}_k) \\ \text{ou} \\ \left[(\omega = (q, x, \star, \psi')) \text{ e } (\bar{q}_k \notin Q_j) \text{ e } (\bar{q}_k \in Q_k)\right] \\ \exists \omega \in \psi, \exists q \in Q_j, \exists x \in \Sigma, \exists \psi' \in \Omega^\star\end{gathered} \tag{3}$$

$$\begin{gathered}(q, x, \star, \psi') \in \psi \rightarrow q' \in Q_k \\ \exists q \in Q_j, \exists x \in \Sigma, \exists \psi' \in \Omega^\star, \exists q' \notin Q_j\end{gathered} \tag{4}$$

$$Q_j \subseteq Q_k \tag{5}$$

Concluímos assim que, uma cadeia w é reconhecida por um $AFAD$ se, e somente se, através de movimentos sucessivos é possível esgotar a cadeia e chegar a um estado final, ou seja, $(Q_j, q_j, w_j, \delta_j, \bar{q}_j) \vdash^\star (Q_k, q_k, \varepsilon, \delta_k, \bar{q}_k)$ e $q_k \in F$ (ε é a cadeia de comprimento zero).

O exemplo 1 demonstra a utilização dessa formulação para a construção de um autômato que reconhece uma linguagem simples.

Exemplo 1: Autômato Adaptativo que reconhece todas as cadeias que contém a mesma quantidade de a's e b's, com nenhum b antes do a.

$$AFAD_1 = (Q_0, \Sigma, \delta_0, q_0, F)$$

$$Q_0 = \{q_0, q_1\}$$

$$\Sigma = \{a, b\}$$

$$\delta_0 = \{(q_0, a, q_1, \psi), (q_1, b, q_0, \emptyset)\}$$

$$F = \{q_0\}$$

$$\psi = \{(\alpha, a, \star, \psi), (\star, b, \alpha, \emptyset)\}$$

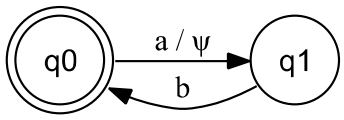

Fig. 2. Diagrama do AFAD do Exemplo 1.

IV. Análise comparativa

A fim de comparar a representação da definição aqui apresentada, serão descritas, a fim de comparação, algumas outras formas de representação de Autômatos Adaptativos.

Seja a linguagem L_1 definida por $L_1 = \{x^{2n}y^{2n}e | n \geq 0\} \cup \{x^{2n+1}y^{2n+1}o | n \geq 0\}$, para tanto, descrevemos um autômato adaptativo AD, representado na forma de um dispositivo adaptativo dirigido por regras [2], [10], que reconhece esta linguagem[16] (figura 3a):

$$AD_0 = (C_0, AR_0, S, c_0, A, NA, BA, AA)$$

$$C_0 = ((1, w), (3, w))$$

$$AR_0 = \left\{\begin{gathered}\left[\left(\varepsilon, c_i, x, c_j, \varepsilon, A(e, o)\right) | c_i = (1, xw) \text{ e } c_j = (1, w)\right], \\ \left[(\varepsilon, c_i, e, c_j, \varepsilon, \varepsilon) | c_i = (1, ew) \text{ e } c_j = (3, w)\right]\end{gathered}\right\}$$

$$S = \{x, y, e, o\}$$

[16] Algumas características, como a linguagem, o formato do autômato e o nome dos estados, foram mantidas inalteradas, conforme descrição original em [10].

$$c_0 = (1, w)$$

$$A = \{(1, \varepsilon), (3, \varepsilon)\}$$

$$NA = BA = \{\varepsilon\}$$

$$AA = \{\varepsilon, A(m,n)\}$$

$$A(m,n) = f, g * \begin{Bmatrix} -[(f,m) \to 3], \\ +[(f,y) \to g], \\ +[(g,n) \to 3], \\ -[(1,x) \to 1, A(m,n)], \\ +[(1,x) \to 1, A(n,m)] \end{Bmatrix}$$

Por motivos práticos, esta descrição foi resumida definindo a cadeia w sobre o alfabeto S. Portanto, todas as ocorrências sobre este símbolo devem ser interpretadas como sendo válidas $\forall w \in S^{\star}$.

Analogamente aos autômatos finitos adaptativos determinísticos, a cada passo k, os estados são representados pelas configurações possíveis C_k e as transições pelo conjunto de regras adaptativas $AR_k = BA \times C \times S \times C \times NA \times AA$, que permitem que o dispositivo modifique a sua configuração e altere o conjunto de regras. Deste modo, partindo de uma configuração inicial c_0, aplicam-se as regras sucessivamente até que o dispositivo atinja uma configuração de aceitação $c \in A$. A configuração deste dispositivo é baseada na do dispositivo subjacente que, neste caso, é o de um Autômato Finito, ou seja, o par (q, w) representando o estado atual e o conteúdo não lido da cadeia de entrada, respectivamente.

Cada vez que uma regra é aplicada, um conjunto de ações adaptativas pode ser aplicado através da execução de uma função adaptativa. Esta função pode ser executada antes e/ou depois de modificar a configuração do dispositivo. Neste caso, o efeito da execução das ações adaptativas se reflete sobre o conjunto de regras adaptativas e diz-se que o dispositivo avançou para o passo seguinte.

O grande interesse aqui reside no conjunto de possíveis funções adaptativas a serem executadas após a modificação da configuração, definido pelo conjunto AA. Neste exemplo, existe uma função vazia (ε) e outra função chamada A, com dois parâmetros.

O conteúdo da função A define a sequência de ações adaptativas a serem executadas. Dentro do seu escopo, também são definidas variáveis locais (f), utilizadas para armazenar a referência a estados/símbolos consultados; e geradores ($g*$), utilizados para armazenar a referência aos estados criados.

Por simplicidade, aqui foram descritas apenas as ações "+" e "−", que representam as ações de inserção e remoção de estados, respectivamente. Além destas, existe também a função de consulta, que fica subentendida por ser a responsável por resolver todas as variáveis não definidas nas chamadas das ações "+" e "−".

É importante notar que, para a descrição se adequar a formulação proposta, são utilizadas apenas as funções adaptativas que se executam após a modificação da configuração do autômato, pois os $AFAD$s sempre executam as ações adaptativas após efetuar a transição entre os estados.

Segundo Pistori [11], "a utilização exclusiva de ações adaptativas anteriores não prejudica a capacidade expressiva do modelo resultante, pois existem artifícios, usando transições vazias, que podem ser empregados na simulação de ações adaptativas posteriores". Como a mesma afirmação se aplica de modo inverso, isto é, transições em vazio podem ser inseridas para a simulação de ações anteriores, esta adaptação não traz nenhuma implicação.

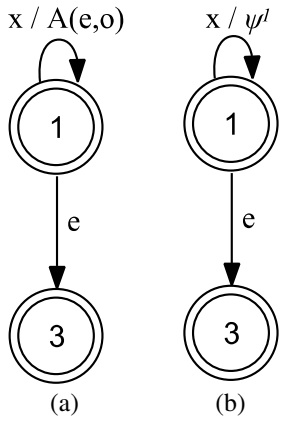

Fig. 3. Diagrama das duas formulações de autômatos apresentadas para o reconhecimento da linguagem L_1. (a) é o diagrama do dispositivo baseado em regras e (b) é do autômato finito adaptativo determinístico.

Os conjuntos NA e BA representam o conjunto de símbolos de saída e o conjunto de funções adaptativas disponíveis para serem executadas antes da mudança de configuração da máquina. Neste exemplo, ambas contém valores nulos devido ao fato deste estudo não possuir nenhum interesse específico nessas funções.

Para servir de parâmetro à comparação, descrevemos a seguir um Autômato Finito Adaptativo Determinístico $AFAD_2$ que também reconhece a linguagem L (figura 3b):

$$AFAD_2 = (Q_0, \Sigma, \delta_0, 1, F)$$

$$Q_0 = \{1,3\}$$

$$\Sigma = \{x, y, e, o\}$$

$$\delta_0 = \{(1,x) \to (1, \psi^1), (1,e) \to (3, \emptyset)\}$$

$$F = \{1,3\}$$

$$\psi^1 = \begin{Bmatrix} (\star, e, d, \emptyset), \\ (\star, y, \star, \emptyset), \\ (\star, o, 3, \emptyset), \\ (1, x, 1, \psi^2) \end{Bmatrix} \quad \psi^2 = \begin{Bmatrix} (\star, o, d, \emptyset), \\ (\star, y, \star, \emptyset), \\ (\star, e, 3, \emptyset), \\ (1, x, 1, \psi^1) \end{Bmatrix}$$

Como se pode notar, as descrições sobre os autômatos finitos adaptativos determinísticos compartilham uma série de características comuns com a representação dirigida a regras

utilizada para representar o Autômato Adaptativo:

- O alfabeto Σ do $AFAD$ é equivalente ao conjunto S de estímulos válidos do AD;
- O conjunto de estados Q_0 é semelhante ao conjunto de configurações válidas C_0;
- O estado inicial q_0 é semelhante à configuração inicial c_0.
- O conjunto de estados finais F é semelhante ao conjunto de configurações de aceitação A.

Uma das diferenças entre esses modelos apresentados está no fato dos dispositivos baseados em regras utilizarem um contador iniciado em zero e que incrementa apenas quando uma ação adaptativa é executada, isto é, a cada alteração, enquanto o $AFAD$ proposto aqui se baseia em uma variável temporal que incrementa sempre que uma transição de estados é efetuada pelo autômato.

Logicamente, esta diferença não exerce grande influência, uma vez que, se um $AFAD$ passa de uma configuração c_i para a configuração c_j sem executar nenhuma ação adaptativa, o conjunto de estados Q_i e a função de transição δ_i permanecem inalterados na nova configuração e apenas o símbolo correspondente da cadeia de entrada é consumido. Com base nisso, pode-se estabelecer uma equivalência entre os modelos considerando que todos os autômatos presentes entre duas ações adaptativas de um $AFAD$ corresponde a um passo do contador do dispositivo dirigido a regras.

Já nas ações adaptativas do $AFAD$, nota-se uma facilidade em relação às do dispositivo dirigido por regras, pois, através do uso do símbolo especial "$\star$" nas mesmas, ele faz com que a variável que armazena o estado consultado e o estado gerado não sejam necessárias.

Na prática, enquanto o dispositivo dirigido por regras implementa o conceito de variáveis locais associadas às funções adaptativas, a definição do $AFAD$ impõe que o último elemento de sua configuração armazene o conteúdo do último estado gerado (ou o estado inicial, caso nenhum tenha sido gerado ainda), funcionando como uma espécie de variável global.

Com relação ao diagrama dos dois autômatos (representados na figura 3), não é possível notar uma grande diferença, uma vez que o diagrama apresenta apenas o modo como os estados estão relacionados entre si, que é a mesma para ambos os autômatos.

V. Conclusão

Este trabalho apresentou uma definição simplificada para a representação de Autômatos Adaptativos.

Dentre as vantagens dessa representação, destacam-se:

- Modelagem simplificada bastante semelhante com a definição dos Autômatos Finitos;
- Inexistência, por definição, de não-determinismo (embora essa ainda possa ser facilmente acrescentada ao modelo);
- Disponibilidade de um método para referência dos estados atual e recém-criado;
- Possibilidade de recursão das ações adaptativas.

Dadas essas características, espera-se que este trabalho colabore para que o desenvolvimento de sistemas envolvendo Autômatos Finitos Adaptativos ocorra de uma maneira mais fácil, prática e intuitiva.

Em trabalhos futuros espera-se explorar com mais detalhes sobre as características da notação proposta a fim de compará-las com outros modelos de computação adaptativa compatíveis, assim como estudar suas propriedades e suas limitações.

Referências

[1] Bó. I. G. L.; Sichman, J. S. Strategy representation complexity in an evolutionary n-Players Prisoner's Dilemma model. In: SEMINARIO DE COMPUTACAO, 16. Blumenau. SC. Anais. Blumenau: 2007. p. 112-124.

[2] Neto, J. J. Adaptive Rule-Driven Devices - General Formulation and Case Study. Lecture Notes in Computer Science. Watson, B.W. and Wood, D. (Eds.): Implementation and Application of Automata 6th International Conference, CIAA 2001, Vol.2494, Pretoria, South Africa, July 23-25, Springer-Verlag, 2001, pp. 234-250.

[3] Neto, J. J. Contribuições à metodologia de construção de compiladores. Tese de Livre Docência, EPUSP, São Paulo, 1993.

[4] Rocha, R. L. A. e Neto, J. J. Autômato adaptativo, limites e complexidade em comparação com máquina de Turing. In: Proceedings of the second Congress of Logic Applied to Technology - LAPTEC 2000. São Paulo: Faculdade SENAC de Ciências Exatas e Tecnologia, p. 33-48, 2001.

[5] Turing, A. M. On Computable Numbers, with an Application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, vol. s2-42. p. 230–265. 1936. [doi:10.1112/plms/s2-42.1.230] and; Turing, A. M.. On Computable Numbers, with an Application to the Entscheidungsproblem: A correction. Proceedings of the London Mathematical Society. vol. s2-43. p. 544–546. 1937. [doi:10.1112/plms/s2-43.6.544].

[6] Sipser, M. Introdução à Teoria da Computação: Tradução da 2ª edição norte-americana (trad. Ruy José Guerra Barreto de Queiroz) Thomson Learning, 2007.

[7] Lewis, H. R. & Papadimitriou, C. H. Elementos de Teoria da Computação (trad. Edson Furmankiewicz) Bookman, 2000.

[8] Cormen, T. H.; Leiserson, C. E.; Rivest, R. L. & Stein, C. Introduction to Algorithms MIT Press, 2002.

[9] Castro JR., A. A.; Neto, J. J.; Pistori, H. Determinismo em Autômatos de Estados Finitos Adaptativos. Revista IEEE América Latina. Vol. 5, Num. 7, ISSN: 1548-0992, Novembro 2007. (p. 515-521).

[10] Ramos, M. V. M.; Neto, J. J. & Vega, Í. S. Linguagens Formais: Teoria, Modelagem e Implementação. Porto Alegre, Bookman, 2009, 656p.

[11] Pistori H.; Neto J. J. Tecnologia adaptativa em engenhara de computação: Estado da arte e aplicações, Tese de Doutorado, Universidade de São Paulo, São Paulo, Brasil, 2003.

[12] Pariente, C. A. B. Gramáticas Adaptativas - Estado Atual da Pesquisa Aplicada em Autômatos Adaptativos. Revista IEEE América Latina. Vol. 5, Num. 7, ISSN: 1548-0992, Novembro 2007. (p. 506-514)

[13] Hromkovic, J.; Karhumaki, J.; Klauck, H.; Schnitger, G.; Seibert, S. Measures of nondeterminism in finite automata, in Proc. ICALP 2000, Lecture Notes in Computer Science, vol. 1853, 2000, pp. 199-210.

[14] Stearns, R. E. Deterministic versus nondeterministic time and lower bound problems. Journal of the ACM, vol. 50, no. 1, pp. 91-95, 2003.