Curso de Engenharia de Computação
ECM253 – Linguagens Formais, Autômatos e Compiladores

Modelos de computação – Autômatos de Pilha

Slides da disciplina ECM253 – Linguagens Formais, Autômatos e Compiladores
Curso de Engenharia de Computação
Instituto Mauá de Tecnologia – Escola de Engenharia Mauá
Prof. Marco Antonio Furlan de Souza
<marco.furlan@maua.br>

Agenda

- Autômato de pilha
- Reconhecimento de cadeias
- Não-determinismo em autômatos de pilha
- Gramáticas livres de contexto

Agenda

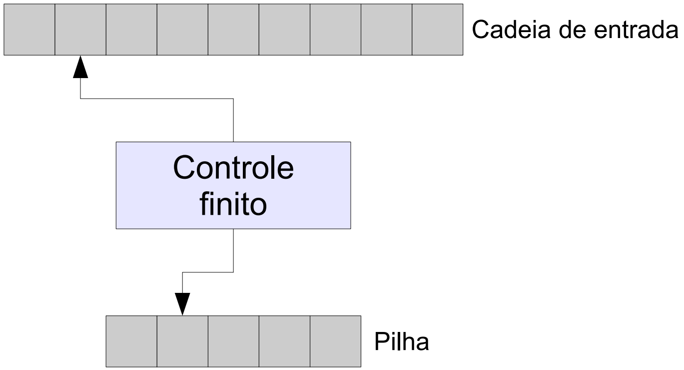
- Autômato de pilha
- Reconhecimento de cadeias
- Não-determinismo em autômatos de pilha
- Gramáticas livres de contexto

Autômato de pilha

- **Pilha**
  - Permite **armazenar dados** utilizando a política “o **último** elemento que **entra** é o **primeiro** que **sai**” (**LIFO**);
  - É uma estrutura **amplamente utilizada** em Computação:
    - Em **hardware**, como **área de armazenamento** substituindo registradores, por exemplo;
    - Em **software**, como **estrutura de dados** servindo como forma de armazenamento para endereços de retorno de chamadas de função, armazenar variáveis locais e parâmetros, traduzir expressões aritméticas pós-fixadas, para citar algumas aplicações.
    - E **autômatos não determinísticos de pilha** utilizam pilha para obter as seguintes propriedades:
      - Reconhecer **linguagens livres de contexto** (o tipo mais empregado em linguagens de programação);
      - **Verificar** de uma cadeia em particular pode ou não ser gerada por uma linguagem livre de contexto.

Autômato de pilha

- **Definição**
  - É uma **máquina de estados finitos** ampliada com uma **memória externa** tipo **pilha**:
    - Um **arquivo** ou **fita infinita** – a **pilha** – para armazenar símbolos durante uma computação;
    - Uma **cabeça** de **leitura/gravação** para **manipular** a **pilha**.



Autômato de pilha

- **Formalização**
  - Um **autômato de pilha** é uma **sêxtupla** $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ onde:
    - Q é um **conjunto** finito de **estados**;
    - Σ é um **conjunto finito** representando o **alfabeto de entrada**;
    - Γ é um **conjunto finito** representando o **alfabeto da pilha**;
    - q_0 é o **estado inicial**;
    - $F \subseteq Q$ é o **conjunto** de **estados finais**;
    - δ é a **função de transição**: $Q \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \rightarrow \wp(Q \times (\Gamma \cup \{\lambda\}))$;
    - λ é a cadeia vazia.
  - A pilha será representada por uma **cadeia** de **elementos da pilha**: o **símbolo** mais à **esquerda** será o elemento do **topo da pilha**;
  - Convencionar-se-á que **símbolos da pilha** serão escritos como **letras maiúsculas** e que **letras gregas** representarão **cadeias** de **símbolos de pilha**. Assim $A\alpha$ é um exemplo de conteúdo de pilha (A no topo acima da cadeia restante).

Autômato de pilha

- **Dinâmica**
  - O autômato consulta o **estado atual**, o **símbolo de entrada**, e o **símbolo no topo da pilha** para **determinar** a **transição** da máquina;
  - Um autômato de pilha é um autômato **não determinístico.**
  - A **função de transição** δ lista as possíveis **transições** para a **combinação** de **estado**, **entrada** e **topo** da pilha **para** um **conjunto** de duplas de **estado** e **topo** da pilha. **Exemplo**:

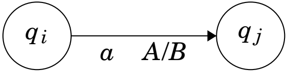
$$\delta(q_i, a, A) = \{(q_j, B), (q_k, C)\}$$

Autômato de pilha

- **Dinâmica**
  - A **transição**:

$$(q_j, B) \in \delta(q_i, a, A)$$

  - Faz com que a máquina:
    1. Altere o estado de q_i para q_j;
    2. Processe o símbolo a (avança a entrada);
    3. Remova o símbolo A do topo da pilha (*pop*);
    4. Adicione o símbolo B no topo da pilha (*push*).
  - **Diagrama de estados** para autômatos de pilha:



Agenda

- Autômato de pilha
- Reconhecimento de cadeias
- Não-determinismo em autômatos de pilha
- Gramáticas livres de contexto

Reconhecimento de cadeias

- **Configuração de um autômato de pilha**
  - A **configuração** de um autômato de pilha é a **tripla** (q_i, w, α) onde:
    - q_i é o **estado atual** da máquina;
    - w é a cadeia de entrada **ainda não processada**;
    - α é a pilha.
  - A **notação**:

$$(q_i, w, \alpha) \vdash_M (q_j, v, \beta)$$

    indica que a configuração (q_j, v, β) pode ser obtida de (q_i, w, α) por uma simples transição do autômato M (o índice M pode ser omitido se não confundir);
  - O símbolo $\vdash_M^*$, por sua vez, representa o **resultado de uma sequência de transições**;
  - Uma **computação** de um autômato de pilha é uma sequência de transições executadas a partir de um estado inicial com a pilha vazia.

Reconhecimento de cadeias

Definição. *Seja* $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ *um* ***autômato de pilha****. Uma cadeia* $w \in \Sigma^*$ *é* ***aceita*** *por* M *se existe uma* ***computação****:*

$$(q_0, w, \lambda) \vdash^* (q_i, \lambda, \lambda)$$

Onde $q_i \in F$. *A* ***linguagem de*** M, $L(M)$ *é o* ***conjunto de cadeias aceitas*** *por* M.

Uma ***computação bem sucedida*** *é aquela que processa* ***toda entrada*** *e* ***para*** *em um* ***estado final*** *com a* ***pilha vazia****.*

Uma ***computação mal sucedida*** *é aquela que processa* ***toda entrada*** *e* ***para*** *em uma* ***configuração diferente*** *daquela de* ***aceitação****.*

Reconhecimento de cadeias

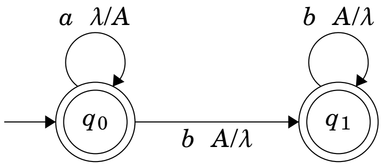
- **Exemplo 1**
  - Autômato de pilha para **reconhecer** a **linguagem** $\{a^i b^i | i \geq 0\}$.
    - Esta **linguagem não é regular**, pois necessitaria de um autômato determinístico de estados infinitos;
    - A solução com autômato de pilha possui em dois estados: o primeiro coleta e empilha informações para controlar i símbolos do tipo a entradas e o segundo é alcançado assim que um símbolo do tipo b é lido, quando, então passa a aceitar somente este tipo de símbolo e desempilhar cada informação correspondente aos símbolos a lidos.

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$
$$Q = \{q_0, q_1\}$$
$$\Sigma = \{a, b\}$$
$$\Gamma = \{A\}$$

$$F = \{q_0, q_1\}$$
$$\delta(q_0, a, \lambda) = \{(q_0, A)\}$$
$$\delta(q_0, b, A) = \{(q_1, \lambda)\}$$
$$\delta(q_1, b, A) = \{(q_1, \lambda)\}$$

Reconhecimento de cadeias

- **Exemplo 1**
  - Diagrama de máquina de estados e exemplo de configurações:



$$
\begin{aligned}
(q_o, aabb, \lambda) &\vdash (q_0, abb, A) \\
&\vdash (q_0, bb, AA) \\
&\vdash (q_1, b, A) \\
&\vdash (q_1, \lambda, \lambda)
\end{aligned}
$$

Reconhecimento de cadeias

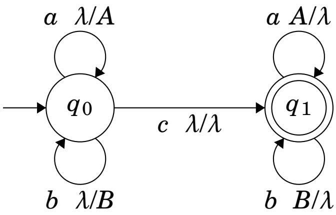
- **Exemplo 2**
  - Autômato de pilha para **reconhecer** a **linguagem** $\{wcw^R | w \in \{a,b\}^*\}$.
  - Neste caso, a pilha será usada para registrar a cadeia w, conforme será processada. Símbolos de pilha A e B representam as entradas a e b, respectivamente.

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$
$$Q = \{q_0, q_1\}$$
$$\Sigma = \{a, b, c\}$$
$$\Gamma = \{A, B\}$$
$$F = \{q_1\}$$

$$\delta(q_0, a, \lambda) = \{(q_0, A)\}$$
$$\delta(q_0, b, \lambda) = \{(q_0, B)\}$$
$$\delta(q_0, c, \lambda) = \{(q_1, \lambda)\}$$
$$\delta(q_1, a, A) = \{(q_1, \lambda)\}$$
$$\delta(q_1, b, B) = \{(q_1, \lambda)\}$$

Reconhecimento de cadeias

- **Exemplo 2**
  - Diagrama de máquina de estados e exemplo de configurações:



$$
\begin{aligned}
(q_0, abcba, \lambda) &\vdash (q_0, bcba, A) \\
&\vdash (q_0, cba, BA) \\
&\vdash (q_1, ba, BA) \\
&\vdash (q_1, a, A) \\
&\vdash (q_1, \lambda, \lambda)
\end{aligned}
$$

Agenda

- Autômato de pilha
- Reconhecimento de cadeias
- Não-determinismo em autômatos de pilha
- Gramáticas livres de contexto

Não-determinismo em autômatos de pilha

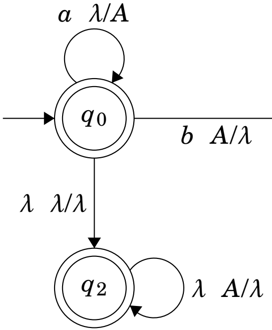
- **Conceitos**
  - Embora a formulação apresentada para autômatos de pilha seja geralmente não-determinística, os **Exemplos 1 e 2** apresentados são exemplos de autômatos de pilha **determinísticos**;
  - Um **autômato de pilha determinístico** é aquele que existe no **máximo uma transição** que é **aplicável** para cada **combinação** de **estado**, **entrada** e **topo da pilha**;
  - Formalmente, é a **condição de compatibilidade** que separa autômatos determinísticos de não-determinísticos:

    Definição. *Duas transições* $(q_j, C) \in \delta(q_i, u, A)$ *e* $(q_k, D) \in \delta(q_i, v, B)$ *são ditas* ***compatíveis*** *se uma das condições a seguir forem satisfeitas:(i)* $u = v$ *e* $A = B$*; (ii)* $u = v$ *e* $A = \lambda$ *ou* $B = \lambda$*; (iii)* $A = B$ *e* $u = \lambda$ *ou* $v = \lambda$*; (iv)* $u = \lambda$ *ou* $v = \lambda$ *e* $A = \lambda$ *ou* $B = \lambda$*.*

  - **Máquinas determinísticas** são aquelas não possuem transições compatíveis distintas.

Não-determinismo em autômatos de pilha

- **Exemplo 3**
  - Pode-se utilizar de não-determinismo para definir autômatos mais “potentes”. Por exemplo, o autômato a seguir reconhece a linguagem $L = \{a^i | i \geq 0\} \cup \{a^i b^i | i \geq 0\}$:



Não-determinismo em autômatos de pilha

- **Exemplo 3**
  - Considerando como exemplo de entrada a cadeia aaa, o autômato testará as configurações a seguir:

(i) **Não aceita**: apesar de q_2 ser final, a cadeia não foi totalmente processada.

$$(q_0, aaa, \lambda) \vdash (q_2, aaa, \lambda)$$

(ii) **Não aceita**: apesar de q_2 ser final, a cadeia não foi totalmente processada.

$$\begin{aligned}(q_0, aaa, \lambda) &\vdash (q_0, aa, A)\\ &\vdash (q_2, aa, A)\\ &\vdash (q_2, aa, \lambda)\end{aligned}$$

(iii) **Não aceita**: apesar de q_2 ser final, a cadeia não foi totalmente processada.

$$\begin{aligned}(q_0, aaa, \lambda) &\vdash (q_0, aa, A)\\ &\vdash (q_0, a, AA)\\ &\vdash (q_2, a, AA)\\ &\vdash (q_2, a, A)\\ &\vdash (q_2, a, \lambda)\end{aligned}$$

Não-determinismo em autômatos de pilha

- **Exemplo 3**
  - Considerando como exemplo de entrada a cadeia aaa, o autômato testará as configurações a seguir (cont.):

    (iv) **Aceita**: q_2 é final e tanto a cadeia restante quanto a pilha estão vazias.

$$
\begin{aligned}
(q_0, aaa, \lambda) &\vdash (q_0, aa, A)\\
&\vdash (q_0, a, AA)\\
&\vdash (q_0, \lambda, AAA)\\
&\vdash (q_2, \lambda, AAA)\\
&\vdash (q_2, \lambda, AA)\\
&\vdash (q_2, \lambda, A)\\
&\vdash (q_2, \lambda, \lambda)
\end{aligned}
$$

Não-determinismo em autômatos de pilha

- **Exemplo 3**
  - Considerando como exemplo de entrada a cadeia $aabb$, o autômato testará as configurações a seguir:

(i) **Não aceita**: apesar de q_2 ser final, a cadeia não foi totalmente processada.

$$(q_0, aabb, \lambda) \vdash (q_2, aabb, \lambda)$$

(ii) **Não aceita**: apesar de q_2 ser final, a cadeia não foi totalmente processada.

$$\begin{aligned}(q_0, aabb, \lambda) &\vdash (q_0, abb, A)\\ &\vdash (q_2, abb, A)\\ &\vdash (q_2, abb, \lambda)\end{aligned}$$

(iii) **Não aceita**: apesar de q_2 ser final, a cadeia não foi totalmente processada.

$$\begin{aligned}(q_0, aabb, \lambda) &\vdash (q_0, abb, A)\\ &\vdash (q_0, bb, AA)\\ &\vdash (q_2, bb, AA)\\ &\vdash (q_2, bb, A)\\ &\vdash (q_2, bb, \lambda)\end{aligned}$$

Não-determinismo em autômatos de pilha

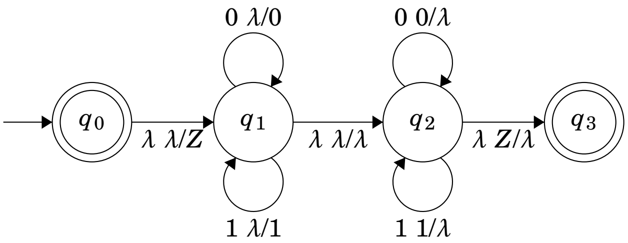
- **Exemplo 3**
  - Considerando como exemplo de entrada a cadeia $aabb$, o autômato testará as configurações a seguir (cont.):

    (iv) **Aceita**: q_2 é final e tanto a cadeia restante quanto a pilha estão vazias.

$$
\begin{aligned}
(q_0, aabb, \lambda) &\vdash (q_0, abb, A) \\
&\vdash (q_0, bb, AA) \\
&\vdash (q_1, b, A) \\
&\vdash (q_1, \lambda, \lambda)
\end{aligned}
$$

Não-determinismo em autômatos de pilha

- **Exemplo 4**
  - O autômato de pilha não-determinístico a seguir reconhece palíndromos pares, $L(M) = \{ww^R | w \in \{0,1\}^*\}$:



  - **Teste seus conhecimentos**: descrever as possíveis configurações que o autômato apresentado deveria testar para encontrar aquela que reconhece a cadeia 011110.

Não-determinismo em autômatos de pilha

- **Observações**
  - Diferentemente de **autômatos finitos**, onde havia uma **equivalência** entre **autômatos determinísticos** e **não-determinísticos**, isso **não ocorre** em **autômatos** finitos de **pilha**;
  - Por exemplo, **não há um autômato determinístico** que aceite a linguagem $L(M) = \{ww^R | w \in \{a,b\}^*$:
    - Como a leitura da cadeia de entrada é da esquerda para a direita, não há como determinar que a primeira metade da cadeia foi lida;
    - Assim, um autômato de pilha tenta "adivinhar" qual é a sequência de configurações que leva a um estado de aceitação, se este existir;
    - As **linguagens aceitas** por um **autômato de pilha** incluem todas as **linguagens regulares**, que é um subconjunto próprio das **linguagens livres de contexto**.

Agenda

- Autômato de pilha
- Reconhecimento de cadeias
- Não-determinismo em autômatos de pilha
- Gramáticas livres de contexto

Gramáticas livres de contexto

- **Definição**
  - Uma **gramática** live de contexto é uma **quádrupla** (V,Σ,P,S) onde:
    - V é um **conjunto de variáveis** da gramática ou **conjunto de símbolos não-terminais – ou variáveis da gramática**;
    - Σ é o **alfabeto** ou conjunto de símbolos **terminais**;
    - P é um conjunto finito de **regras de produção**;
    - S é um elemento distinto de V denominado **símbolo de partida**.
  - As **regras de produção** de gramáticas livres de contexto são **escritas** como $A \to w$, onde $A \in V$ e $w \in (V \cup \Sigma)^*$ e λ pode ocorrer do lado direito de uma regra. Em outras palavras, são regras que do lado esquerdo só se tem um único símbolo não-terminal e que do lado direito se tem o símbolo λ ou uma cadeia contendo terminais e não-terminais.
  - **Exemplo** de uma gramática $G=(V,\Sigma,P,S)$, livre de contexto:
    - $V=\{E,N\}$
    - $\Sigma=\{\mathtt{0},\mathtt{1},\mathtt{2},\mathtt{3},\mathtt{4},\mathtt{5},\mathtt{6},\mathtt{7},\mathtt{8},\mathtt{9},\mathtt{+},\mathtt{-},\mathtt{*},\mathtt{/},\mathtt{(},\mathtt{)}\}$
    - $S=E$
    - $P=\{E\to N, E\to \mathtt{(}E\mathtt{)}, E\to E\mathtt{+}E, E\to E\mathtt{-}E, E\to E\mathtt{*}E, E\to E\mathtt{/}E\}$

Gramáticas livres de contexto

- **Definição**
  - As **gramáticas livres de contexto são importantes** pois são comumente **utilizadas** para descrever a **sintaxe** da maioria das **linguagens de programação** conhecidas.
  - Uma **gramática** é um **modelo de computação** conhecido como **dispositivo gerador de linguagens** (autômatos são dispositivos reconhecedores de linguagem).

Gramáticas livres de contexto

- **Exemplo**
  - A gramática $G = (V, \Sigma, P, S)$, definida a seguir, **gera** a linguagem $\{a^i b^i | i > 0\}$:

$$
\begin{aligned}
S &\rightarrow aAB|aB \\
A &\rightarrow aAB|aB \\
B &\rightarrow b
\end{aligned}
$$

  - Exemplo de geração de cadeias da linguagem:

$$
\begin{aligned}
S &\rightarrow aAB && \text{(regra de partida)} \\
&\rightarrow aaABB && \text{(aplicação da segunda regra)} \\
&\rightarrow aaaBBB && \text{(aplicação da segunda regra)} \\
&\rightarrow aaabBB && \text{(aplicação da terceira regra)} \\
&\rightarrow aaabbB && \text{(aplicação da terceira regra)} \\
&\rightarrow aaabbb && \text{(aplicação da terceira regra)}
\end{aligned}
$$

Gramáticas livres de contexto

- **Reconhecimento por autômatos de pilhas**
  - Pode-se provar que toda **linguagem livre de contexto** pode ser **aceita** por um **autômato de pilha**;
  - Para isso, as **regras da gramática** são utilizadas para gerar **transições** em um autômato de pilha equivalente;
  - Essa tarefa é facilitada se a gramática estiver em uma **forma normal**. A forma normal de uma gramática define uma gramática equivalente onde são **eliminadas** a **recursão do símbolo de partida**, **regras vazias**, **regras encadeadas** e **símbolos sem uso**;
  - Uma maneira formal de gramática apropriada para ser aceita por um autômato de pilha é a **forma normal de Greibach**, na qual as **produções** possuem sempre a **forma** $A \to aA_1A_2 \ldots A_n$, exceto se a gramática **aceitar** λ, que então se **adiciona** a regra $S \to \lambda$ (S é o símbolo de partida).

Gramáticas livres de contexto

- **Reconhecimento por autômatos de pilhas**
  - Um **autômato de pilha estendido** é aquele que, em sua operação, **empilha** mais de um símbolo por vez na sua pilha;
  - Por **exemplo**, a transição $(q_j, BCD) \in \delta(q_i, a, A)$ empilha os símbolos BCD na pilha, com o símbolo B sendo seu novo topo;
  - A ideia é, em uma **derivação mais à esquerda**, empilhar as variáveis $A_1 A_2 \ldots A_n$ de uma regra de produção e então processá-las da esquerda para direita;
  - O **autômato equivalente** tem apenas **dois estados**: estado de **início** q_0 e estado de **aceitação** q_1;
  - Uma regra $S \to aA_1 A_2 \ldots A_n$ gera uma transição que **processa o símbolo terminal** a, empilha as variáveis $A_1 A_2 \ldots A_n$ na pilha e entra no estado q_1;
  - O restante da computação utiliza o símbolo de entrada e o topo da pilha para determinar a transição apropriada.

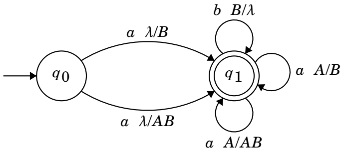
Gramáticas livres de contexto

- **Exemplo de reconhecimento**
  - A gramática $G = (V, \Sigma, P, S)$, definida a seguir, aceita a linguagem $\{a^i b^i | i > 0\}$:

$$S \rightarrow aAB | aB$$
$$A \rightarrow aAB | aB$$
$$B \rightarrow b$$

  - A função de transição do autômato de pilha equivalente é obtida diretamente das regras de G:

$$\delta(q_0, a, \lambda) = \{(q_1, AB), (q_1, B)\}$$
$$\delta(q_1, a, A) = \{(q_1, AB), (q_1, B)\}$$
$$\delta(q_1, b, B) = \{(q_1, \lambda)\}$$



Gramáticas livres de contexto

- **Exemplo de reconhecimento**
  - Considerando como **cadeia de entrada** $aaabbb$, tem-se a seguinte computação, onde, na esquerda, é apresentada as derivações pela gramática e, na direita, as computações do autômato:

$$
\begin{aligned}
S &\Rightarrow aAB & (q_0, aaabbb, \lambda) &\vdash (q_1, aabbb, AB) \\
&\Rightarrow aaABB & &\vdash (q_1, abbb, ABB) \\
&\Rightarrow aaaBBB & &\vdash (q_1, bbb, BBB) \\
&\Rightarrow aaabBB & &\vdash (q_1, bb, BB) \\
&\Rightarrow aaabbB & &\vdash (q_1, b, B) \\
&\Rightarrow aaabbb & &\vdash (q_1, \lambda, \lambda)
\end{aligned}
$$

Teste seus conhecimentos

(1) Seja o autômato de pilha M definido por:

$$Q = \{q_0, q_1, q_2\}$$
$$\Sigma = \{a, b\}$$
$$\Gamma = \{A\}$$
$$F = \{q_1, q_2\}$$

$$\delta(q_0, a, \lambda) = \{(q_0, A)\}$$
$$\delta(q_0, \lambda, \lambda) = \{(q_1, \lambda)\}$$
$$\delta(q_0, b, A) = \{(q_2, \lambda)\}$$
$$\delta(q_1, \lambda, A) = \{(q_1, \lambda)\}$$
$$\delta(q_2, b, A) = \{(q_2, \lambda)\}$$
$$\delta(q_2, \lambda, A) = \{(q_2, \lambda)\}$$

a) Elaborar o diagrama de estados de M.
b) Descrever a linguagem aceita por M.
c) Simular as computações das cadeias aab, abb e aba em M.
d) Provar que $aabb$ e $aaab \in L(M)$.

Teste seus conhecimentos

(2) Construir autômatos de pilha que aceitem as linguagens a seguir:

a) $\{a^i b^j | 0 \leq i \leq j\}$.

b) $\{a^i b^j | i \neq j\}$.

c) $\{w | w \in \{a, b\}^*$ e w possui duas vezes mais as do que bs$\}$.

(3) Construir um autômato de pilha que aceite a linguagem gerada pela gramática a seguir, na forma normal de Greibach:

$$S \to aABA | aBB$$
$$A \to bA | b$$
$$B \to cB | c$$

Referências bibliográficas

GERSTING, J. **Fundamentos Matemáticos para a Ciência da Computação: um Tratamento Moderno de Matemática Discreta**. [S.l.]: Livros Técnicos e Científicos. ISBN 9788521614227.

RICH, E. **Automata, Computability and Complexity: Theory and Applications**. [S.l.]: Pearson Prentice Hall, 2008.

ROSEN, K. **Discrete Mathematics and Its Applications**. New York: McGraw-Hill, 2003. (McGraw-Hill higher education).

SUDKAMP, T. **Languages and Machines: An Introduction to the Theory of Computer Science**. [S.l.]: Pearson Addison-Wesley, 2006.

TAYLOR, R. G. **Models of computation and formal languages**. New York: Oxford University Press, 1998.