Universidade Federal do Maranhão
Centro de Ciências Exatas e Tecnologias
Engenharia da Computação

Thales L. A. Valente

Disciplina: Linguagens Formais e Autômatos
Código: EECP0020

27 de maio de 2024

Conteúdo programático

- Elementos de matemática discreta
- Conceitos básicos de linguagens
- Linguagens regulares e autômatos finitos
- Linguagens livres de contexto e autômatos de pilha
- Linguagens sensíveis ao contexto e Máquinas de Turing com fita limitada
- Linguagens recursivas e Máquinas de Turing com finta infinita
- Linguagens recursivamente enumeráveis

Sumário

Hierarquia de Chomsky

- De acordo com a complexidade relativa das linguagens, Chomsky definiu uma classificação que permite antecipar as propriedades fundamentais das linguagens e vislumbrar os modelos de implementação mais adequados. A **Hierarquia de Chomsky** é ilustrada abaixo:

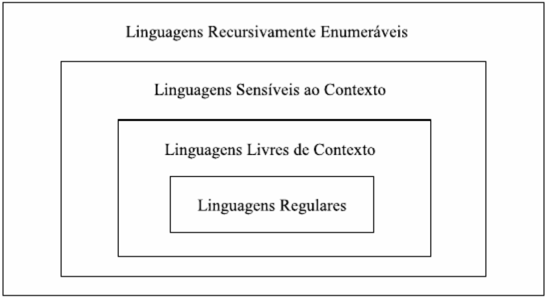


Figura: Hierarquia de Chomsky

Linguagens Livres de Contexto ou do tipo 2

- O estudo das Linguagens Livres de Contexto é importante para a Computação, pois:
  - Consegue tratar de questões como parênteses balanceados, contruções bloco-estruturadas, entre outras, típicas de linguagens de programação como Pascal, C, etc.
  - Os algoritmos que implementam as Linguagens Livres de Contexto são relativamente simples e possuem boa eficiência.
  - São bastante utilizados em analisadores sintáticos, tradutores de linguagens e processadores de texto em geral.

Linguagens livres de contexto ou do tipo 2

- O estudo de Linguagens Livres de Contexto pode ser abordado a partir de dois formalismos básicos:
  - **Operacional ou reconhecedor**: representado por autômatos de pilha.
  - **Axiomático ou gerador**: representado por gramáticas livres de contexto.

Definição

- Uma **Gramática Livre de Contexto (GLC)** G é uma gramática

$$G = (V, T, P, S)$$

  com a restrição de que qualquer regra de produção de P seja da forma $A \rightarrow \alpha$, onde A é um não-terminal (variável) e $\alpha \in (V \cup T)^*$.

- A gramática é dita livre de contexto pelo fato de o não-terminal A estar desacompanhado de outros símbolos (terminais e/ou não-terminais), que definiriam o seu "contexto".
  - Note que toda Gramática Regular é Livre de Contexto.

Definição

- **Exemplo**: Considere a linguagem $\boldsymbol{L_1} = \{a^n b^n | n \geq 0\}$.
- A GLC $\boldsymbol{G_1}$ é tal que $GERA(G_1) = L_1$:

$$G_1 = (\{S\}, \{a, b\}, \{S \to aSb | S \to \epsilon\}, S)$$

- Por exemplo, a palavra *aabb* pode ser gerada pela seguinte derivação:

$$S \implies aSB \implies aaSbb \implies aa\epsilon bb \implies aabb$$

- Por que essa gramática é importante em computação?

Definição

- **Exemplo**: Considere a linguagem $\boldsymbol{L_1} = \{a^n b^n | n \geq 0\}$.
- A GLC $\boldsymbol{G_1}$ é tal que $GERA(G_1) = L_1$:

$$G_1 = (\{S\}, \{a, b\}, \{S \rightarrow aSb | S \rightarrow \epsilon\}, S)$$

- Por exemplo, a palavra *aabb* pode ser gerada pela seguinte derivação:

$$S \implies aSB \implies aaSbb \implies aa\epsilon bb \implies aabb$$

- Por que essa gramática é importante em computação?
  - Porque ela define o **duplo balanceamento** em linguagens bloco-estruturada e em parênteses balanceados.

Definição

- **Exemplo**: Considere a linguagem $\boldsymbol{L_2}$ composta pelas expressões aritméticas contendo colchetes balanceados, dois operadores e um operando.
- A GLC $\boldsymbol{G_2}$ é tal que $GERA(G_2) = L_2$:

$$G_2 = (\{E\}, \{+, *, [,], x\}, P_2, E)$$

$$P_2 = \{E \to E + E | E * E | [E] | x\}$$

- Por exemplo, a palavra $[x + x] * x$ pode ser gerada pela seguinte derivação:

$$E \implies E * E \implies [E] * E \implies [E + E] * E \implies$$

$$[x + E] * E \implies [x + x] * E \implies [x + x] * x$$

- Esta sequência de derivação é única?

Definição

- Uma linnguagem é dita **Linguagem Livre de Contexto (LLC) ou do Tipo 2** se for gerada por uma Gramática Livre de Contexto.

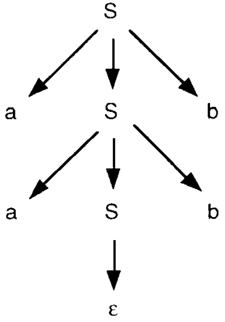
Árvore de Derivação

- Uma maneira alternativa de visualizar a derivação de uma palavra por uma GLC é por meio de **Árvore de Derivação**, definida abaixo:

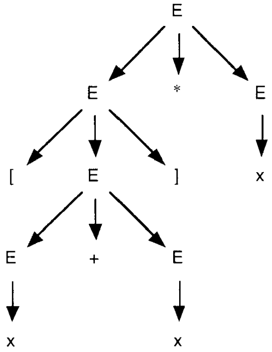
  1. A **raiz** é o símbolo inicial da gramática.
  2. Os **vértices não-folhas** são os não-terminais. Um nó A com filhos X_1, X_2 e X_3 representam a regra $A \rightarrow X_1X_2X_3$.
  3. Os **vértices folha** representam terminais ou o ϵ.

Árvore de Derivação

- **Exemplo**: considere os palavras *aabb* e $[x + x] * x$, dos exemplos anteriores. Suas árvores de derivação são, respectivamente:



(a) Duplo desbalanceamento

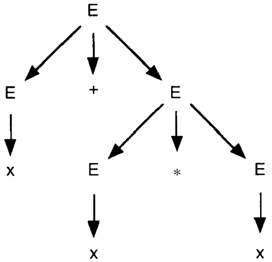


(b) Expressão aritmética

- **Observação**: uma única árvore de derivação pode representar derivações distintas de uma mesma palavra.

Árvore de Derivação

- **Exemplo**: Considere a árvore de derivação abaixo, que representa a derivaçãoo da palavra $x + x * x$:



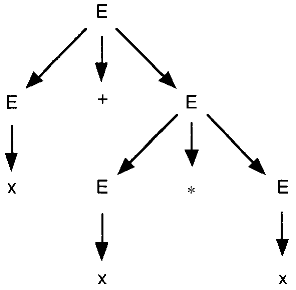
- A palavra pode ser gerada por diversas derivações, como segue:
  1. $E \implies E{+}E \implies x{+}E \implies x{+}E{*}E \implies x{+}x{*}E \implies x{+}x{*}x$
  2. $E \implies E{+}E \implies E{+}E{*}E \implies E{+}E{*}x \implies E{+}x{*}x \implies x{+}x{*}x$
  3. $E \implies E{+}E \implies E{+}E{*}E \implies x{+}E{*}E \implies x{+}x{*}E \implies x{+}x{*}x$
  4. Etc...

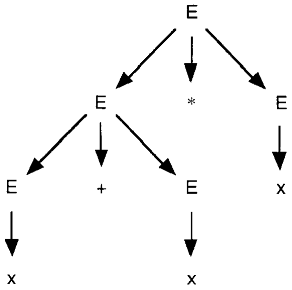
Árvore de Derivação

- **Derivação mais à esquerda** de uma árvore de derivação é a sequência de produção aplicada sempre à variável mais à esquerda.
  - O exemplo 1) anterior é uma derivação mais à esquerda.
- **Derivação mais à direita** de uma árvore de derivação é a sequência de produção aplicada sempre à variável mais à direita.
  - O exemplo 2) anterior é uma derivação mais à direita.

Ambiguidade

- Uma GLC é dita **Gramática Ambígua** se existe uma palavra que possui mais de uma árvore de derivação.
- **Exemplo**: A palavra $x + x * x$ do exemplo anterior pode ser gerada por árvores de derivação distintas, como ilustrado abaixo:





Ambiguidade

- Ainda considerando o exemplo anterior, note que a palavra $x + x * x$ possui mais de uma derivação à esquerda (à direita):
  - Derivação mais à esquerda:
    - $E \implies E+E \implies x+E \implies x+E*E \implies x+x*E \implies x+x*x$
    - $E \implies E*E \implies E+E*E \implies x+E*E \implies x+x*E \implies x+x*x$
  - Derivação mais à direita:
    - $E \implies E+E \implies E+E*E \implies E+E*x \implies E+x*x \implies x+x*x$
    - $E \implies E*E \implies E*x \implies E+E*x \implies E+x*x \implies x+x*x$
- Alternativamente, linguagem ambígua pode ser definida aquele que possui uma palavra com mais de uma derivação mais à esquerda (direita).

Ambiguidade

- Uma linguagem é **inerentemente ambígua** se qualquer GLC que a define é ambígua.
- **Exemplo**: A linguagem abaixo é inerentemente ambígua:

$$\{w|w = a^n b^n c^m d^m \text{ou} a^n b^m c^m d^n, n \geq 1, m \geq 1\}$$

Simplificação de Gramáticas Livres de Contexto

- O material sobre **simplificação de Gramáticas Livres de Contexto** encontra-se em slides próprios disponíveis no SIGAA.

Formas normais

- O formato das regras das GLCs é muito abrangente. Para impor restrições mais rígidas na definição das regras, podem-se utilizar as chamadas **formas normais**. Elas são úteis em:
  - Desenvolvimento de alguns algoritmos reconhecedores de linguagens.
  - Prova de alguns teoremas.
- Há duas forms normais mais utilizadas:
  - **Forma Normal de Chomsky (FNC)**.
  - **Forma Normal de Greibach (FNB)**.

Forma Normal de Chomsky

- Na **FNC**, todas as regras da GLC são da forma:

$$A \to BC \text{ ou } A \to a,$$

com A, B, C variáveis e a terminal.

Forma Normal de Chomsky

- Para transformar uma GLC qualquer para a FNC, basta seguir o algoritmo de três passos:
  - **Passo 1**: Simplificar a gramática.
  - **Passo 2**: Para regras com lado direito maior ou igual a dois: fazer o lado direito ter exclusivamente variáveis.
  - **Passo 3**: Para regras com lado direito maior ou igual a três: fazer o lado direito ter exatamente duas variáveis.

Forma Normal de Chomsky

- **Exemplo**: Transformar a GLC $\boldsymbol{G_2}$ em FNC:

$$\boldsymbol{G_2} = (\{E\}, \{+, *, [,], x\}, P_2, E)$$

$$P_2 = \{E \rightarrow E + E | E * E | [E] | x\}$$

- **Passo 1**: Simplificar a gramática.
  - A gramática já está simplificada.

Forma Normal de Chomsky

- **Exemplo**: Transformar a GLC $\boldsymbol{G_2}$ em FNC:

$$\boldsymbol{G_2} = (\{E\}, \{+, *, [,], x\}, P_2, E)$$

$$P_2 = \{E \rightarrow E + E | E * E | [E] | x\}$$

- **Passo 2**: Para regras com lado direito maior ou igual a dois: fazer o lado direito ter exclusivamente variáveis.
  - Exceto pela regra $E \rightarrow x$, as demais regras devem ser substituídas:

$$E \rightarrow EC_+E | EC_*E | C_[EC_]$$

$$C_+ \rightarrow +$$

$$C_* \rightarrow *$$

$$C_[\rightarrow [$$

$$C_] \rightarrow]$$

Forma Normal de Chomsky

- **Exemplo**: Transformar a GLC $\boldsymbol{G_2}$ em FNC:

$$\boldsymbol{G_2} = (\{E\}, \{+, *, [,], x\}, P_2, E)$$

$$P_2 = \{E \rightarrow E + E | E * E | [E] | x\}$$

- **Passo 3**: Para regras com lado direito maior ou igual a três: fazer o lado direito ter exatamente duas variáveis.
  - As regras $E \rightarrow EC_+E | EC_*E | C_[EC_]$ devem ser substituídas:

$$E \rightarrow ED_1 | ED_2 | C_[D_3$$

$$D_1 \rightarrow C_+E$$

$$D_2 \rightarrow C_*E$$

$$D_3 \rightarrow EC_]$$

Forma Normal de Chomsky

- **Exemplo**: Transformar a GLC $\boldsymbol{G_2}$ em FNC:

$$\boldsymbol{G_2} = (\{E\}, \{+, *, [,], x\}, P_2, E)$$

$$P_2 = \{E \to E + E | E * E | [E] | x\}$$

- A gramática $\boldsymbol{G_2'}$ em FNC fica:

$$\boldsymbol{G_2'} = (\{E, C_+, C_*, C_[, C_], D_1, D_2, D_3\}, \{+, *, [,], x\}, P_2', E)$$

$$P_2' = \{E \to ED_1 | ED_2 | C_[D_3 | x,$$

$$D_1 \to C_+ E, D_2 \to C_* E, D_3 \to EC_],$$

$$C_+ \to +, C_* \to *, C_[\to [, C_] \to]\}$$

Forma Normal de Greibach

- Na **FNG**, todas as regras da GLC são da forma:

$$A \to a\alpha,$$

com A variável, a terminal e α sequência de variáveis.

Forma Normal de Greibach

- Para transformar uma GLC qualquer para a FNG, basta seguir o algoritmo de seis passos:
  - **Passo 1**: Simplificar a gramática.
  - **Passo 2**: Renomear as variáveis em ordem crescente.
  - **Passo 3**: Transformar as regras para a forma $A_r \rightarrow A_s\alpha$, para $r \leq s$.
  - **Passo 4**: Excluir as recursões da forma $A_r \rightarrow A_r\alpha$.
  - **Passo 5**: Fazer cada regra ter um terminal no início do lado direito.
  - **Passo 6**: Transformar todas as regra para a forma $A \rightarrow a\alpha$, com α composto exclusivamente de variáveis.

Forma Normal de Greibach

- **Exemplo**: Transformar a GLC $\boldsymbol{G}$ em FNG:

$$\boldsymbol{G} = (\{S, A\}, \{a, b\}, P, S)$$

$$P = \{S \rightarrow AA|a, A \rightarrow SS|b\}$$

- **Passo 1**: Simplificar a gramática.
  - A gramática já está simplificada.

Forma Normal de Greibach

- **Exemplo**: Transformar a GLC $\boldsymbol{G}$ em FNG:

$$\boldsymbol{G} = (\{S, A\}, \{a, b\}, P, S)$$

$$P = \{S \rightarrow AA|a, A \rightarrow SS|b\}$$

- **Passo 2**: Renomear as variáveis em ordem crescente.
  - As variáveis S, A são renomeadas para A_1, A_2, respectivamente. As regras de G ficam como segue:

$$A_1 \rightarrow A_2A_2|a$$

$$A_2 \rightarrow A_1A_1|b$$

Forma Normal de Greibach

- **Exemplo**: Transformar a GLC $\boldsymbol{G}$ em FNG:

$$\boldsymbol{G} = (\{S, A\}, \{a, b\}, P, S)$$

$$P = \{S \rightarrow AA|a, A \rightarrow SS|b\}$$

- **Passo 3**: Transformar as regras para a forma $A_r \rightarrow A_s\alpha$, para $r \leq s$.
  - A regra $A_2 \rightarrow A_1A_1$ precisa ser modificada. Substitui-se A_1 por suas regras, onde necessário. As regras da gramática ficam como segue:

$$A_1 \rightarrow A_2A_2|a$$

$$A_2 \rightarrow A_2A_2A_1|aA_1|b$$

Forma Normal de Greibach

- **Exemplo**: Transformar a GLC $\boldsymbol{G}$ em FNG:

$$\boldsymbol{G} = (\{S, A\}, \{a, b\}, P, S)$$

$$P = \{S \to AA|a, A \to SS|b\}$$

- **Passo 4**: Excluir as recursões da forma $A_r \to A_r\alpha$.
  - A regra $A_2 \to A_2A_2A_1$ contém uma recursão e precisa ser removida, introduzindo uma variável auxiliar B e incluindo recursão à direita ($B_r \to \alpha B_r$) e acrescentando regras para A_2 finalizando com B. As regras da gramática ficam como segue:

$$A_1 \to A_2A_2|a$$

$$A_2 \to aA_1|b|aA_1B|bB$$

$$B \to A_2A_1|A_2A_1B$$

Forma Normal de Greibach

- **Exemplo**: Transformar a GLC $\boldsymbol{G}$ em FNG:

$$\boldsymbol{G} = (\{S, A\}, \{a, b\}, P, S)$$

$$P = \{S \rightarrow AA|a, A \rightarrow SS|b\}$$

- **Passo 5**: Fazer cada regra ter um terminal no início do lado direito.
  - O lado direito das regras da maior variável A_2 iniciam por um terminal. Substitue-se A_2 nas regras necessárias. As regras da gramática ficam como segue:

$$A_1 \rightarrow aA_1A_2|bA_2|aA_1BA_2|bBA_2|a$$

$$A_2 \rightarrow aA_1|b|aA_1B|bB$$

$$B \rightarrow aA_1A_1|bA_1|aA_1BA_1|bBA_1|aA_1A_1B|bA_1B|aA_1BA_1B|bBA_1B$$

Forma Normal de Greibach

- **Exemplo**: Transformar a GLC $\boldsymbol{G}$ em FNG:

$$\boldsymbol{G} = (\{S, A\}, \{a, b\}, P, S)$$

$$P = \{S \rightarrow AA|a, A \rightarrow SS|b\}$$

- **Passo 6**: Transformar todas as regra para a forma $A \rightarrow a\alpha$, com α composto exclusivamente de variáveis.
  - Para todas as regras, α já possui exclusivamente variáveis.

Forma Normal de Greibach

- **Exemplo**: Transformar a GLC $\boldsymbol{G}$ em FNG:

$$\boldsymbol{G} = (\{S, A\}, \{a, b\}, P, S)$$

$$P = \{S \rightarrow AA|a, A \rightarrow SS|b\}$$

- A gramática $\boldsymbol{G'}$ em FNG fica:

$$\boldsymbol{G'} = (\{A_1, A_2, B\}, \{a, b\}, P', A_1)$$

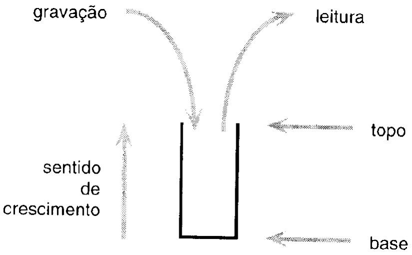
$$\boldsymbol{P'} = \{A_1 \rightarrow aA_1A_2|bA_2|aA_1BA_2|bBA_2|a,$$

$$A_2 \rightarrow aA_1|b|aA_1B|bB,$$

$$B \rightarrow aA_1A_1|bA_1|aA_1BA_1|bBA_1|aA_1A_1B|bA_1B|aA_1BA_1B|bBA_1B\}$$

Autômato com Pilha

- A classe das Linguagens Livres de Contexto pode ser reconhecida por **Autômatos com Pilha**.
  - Possui uma **pilha** como memória auxiliar.
  - Possui a facilidade de **não-determinismo**.



Autômato com Pilha

- Formalmente, um **Autômato com Pilha Não-determinístico (APN)**, ou simplesmente **Autômato com Pilha (AP)** M é uma 6-tupla:

$$M = \{\Sigma, Q, \delta, q_0, F, V\}$$

onde:

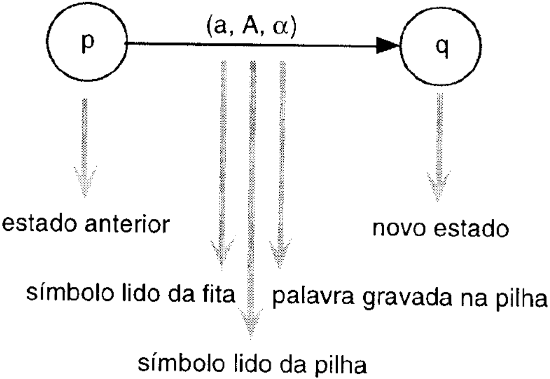
  - Σ: alfabeto de símbolos de entrada.
  - Q: conjunto finito de estados do autômato.
  - δ: função programa da forma:

$$\delta : Q \times (\Sigma \cup p\{\epsilon, ?\}) \times (V \cup \{\epsilon, ?\}) \to 2^{Q \times V^*}$$

  - q_0: estado inicial, tal que $q_0 \in Q$.
  - F: conjunto de estados finais, tal que $F \subset Q$.
  - V: alfabeto da pilha.

Autômato com Pilha

- Graficamente, a função programa pode ser representada como:



Autômato com Pilha

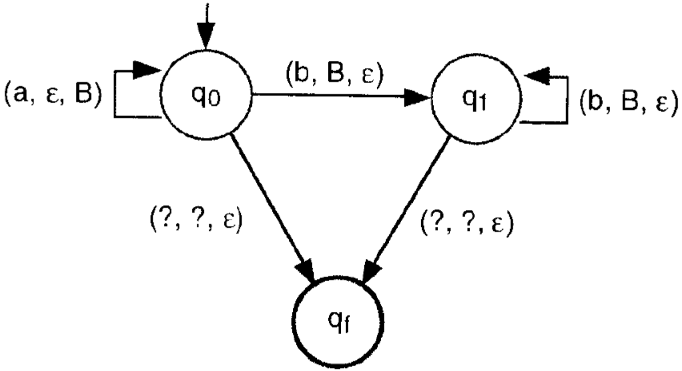
- Algumas observações quanto à função programa podem ser feitas:
  - A função δ pode ser total.
  - O símbolo ϵ na leitura indica a facilidade de não determinismo da fita ou da pilha.
  - O símbolo ϵ na gravação indica que nenhuma gravação é realizada na pilha (e não move a cabeça).
  - O símbolo ? pode indicar, a depender da posição:
    - **Na leitura da fita**: a palavra foi toda lida?
    - **Na leitura da pilha**: a pilha estah vazia?

Autômato com Pilha

- Um Autômato com Pilha pode parar aceitando ou rejeitando a entrada ou ficar em um *loop* infinito, como segue:
  - Um dos autômatos alternativos assume um estado final: o autômato pára e a palavra é aceita;
  - Todos os caminhos alternativos rejeitam a entrada: o autômato pára e a palavra é rejeitada e;
  - Pelo menos um caminho alternativo está e *loop* infinito e o demais rejeitam (ou estão em *loop* infinito também): o autômato está em *loop* infinito.

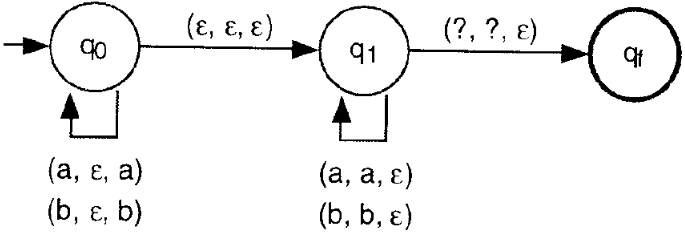
Autômato com Pilha

- **Exemplo**: A linguagem $L = \{a^n b^n | n \geq 0\}$ é reconhecida pelo seguinte autômato:



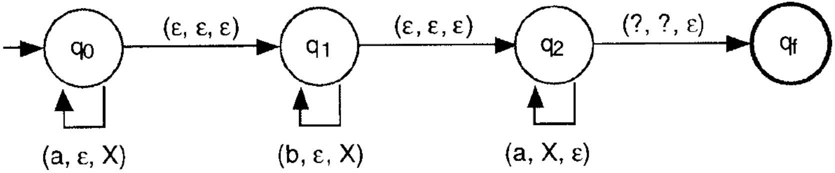
Autômato com Pilha

- **Exemplo**: A linguagem $L = \{ww^r | w \in \{a, b\}^*\}$ é reconhecida pelo seguinte autômato:



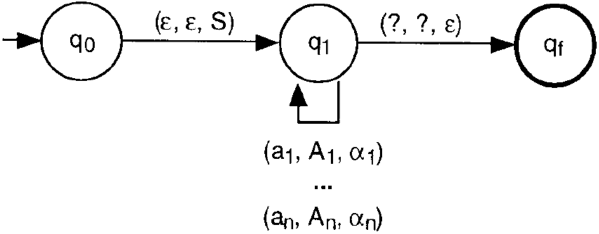
Autômato com Pilha

- **Exemplo**: A linguagem $L = \{a^n b^m a^{n+m}\}$ é reconhecida pelo seguinte autômato:



Autômato com Pilha

- A partir de uma Linguagem Livre de Contexto na Forma Normal de Greibach (regras da forma $A \rightarrow a\alpha$, com α uma sequência de variáveis), pode-se **construir a Autômato com Pilha correspondente** como segue:
  - Seja $G = (V, T, P, S)$ uma GLC na FNG;
  - Seja $M = (T, \{q_0, q_1, q_f\}, \delta, q_0, \{q_f\}, V)$, onde:
    - $\delta(q_0, \epsilon, \epsilon) = \{(q_1, S)\}$
    - $\delta(q_1, a, A) = \{(q_1, \alpha) | A \rightarrow a\alpha \in P\}$
    - $\delta(q_1, ?, ?) = \{(q_f, \epsilon)\}$
- O Autômato com Pilha G resultante é ilustrado abaixo:



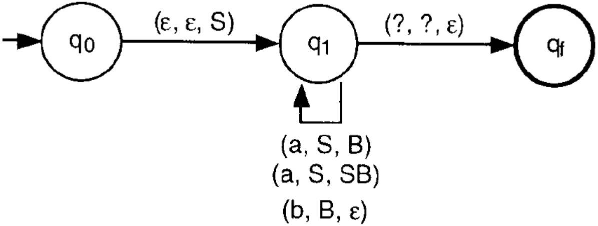
Autômato com Pilha

- **Exemplo**: A linguagem $L = \{a^n b^n | n \geq 1\}$ é gerada pela seguinte gramática:

$$G = (\{S, B\}, \{a, b\}, P, S)$$

$$P = \{S \rightarrow aB | aSB, B \rightarrow b\}$$

  - A linguagem L é reconhecida pelo seguinte autômato:



Bibliografia

1. MENEZES, Paulo B. Linguagens formais e autômatos. 6ª ed. Porto Alegre: Bookman, 2011.
    - **Capítulos 5**.

Dúvidas?