

# **Linguagens Formais e Autômatos**

## **Aula 17 - Autômatos com pilha**

# Referências bibliográficas

- **Introdução à teoria dos autômatos, linguagens e computação / John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman** ; tradução da 2.ed. original de Vandenberg D. de Souza. - Rio de Janeiro : Elsevier, 2002 (Tradução de: Introduction to automata theory, languages, and computation - ISBN 85-352-1072-5)
  - Capítulo 6 - Seções 6.1 e 6.2
- **Introdução à teoria da computação / Michael Sipser** ; tradução técnica Ruy José Guerra Barretto de Queiroz ; revisão técnica Newton José Vieira. -- São Paulo : Thomson Learning, 2007 (Título original : Introduction to the theory of computation. "Tradução da segunda edição norte-americana" - ISBN 978-85-221-0499-4)
  - Capítulo 2 - Seção 2.2

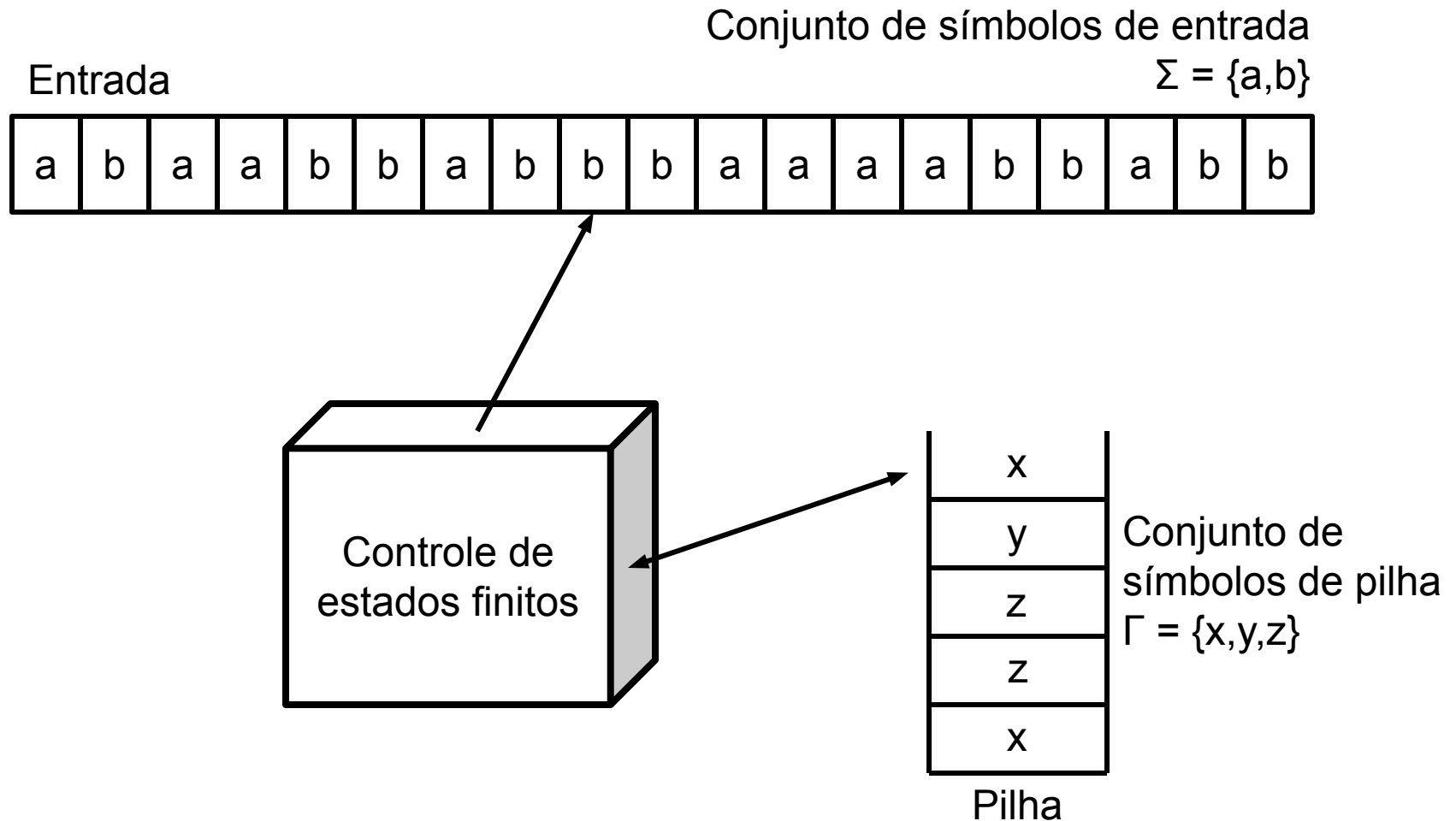
# Autômatos com pilha

- Antes: o que é pilha?
- Estrutura de dados
  - Primeiro-que-entra-é-o-último-que-sai
  - FILO (First In Last Out)
- Como uma pilha de pratos
  - Você sempre pega o prato que está em cima
  - Novos pratos vão para o topo
  - Quando acabarem os pratos (pilha vazia) é hora de lavar a louça

# Autômatos com pilha

- Essencialmente, é um  $\epsilon$ -NFA com a inclusão de uma pilha
- Pilha = memória adicional
  - Além dos estados finitos
  - Quantidade infinita de informações
- A pilha pode ser lida, aumentada e diminuída apenas no topo
- Autômatos com pilha reconhecem todas as linguagens livres de contexto
  - E apenas elas

# Autômatos com pilha



# Autômatos com pilha

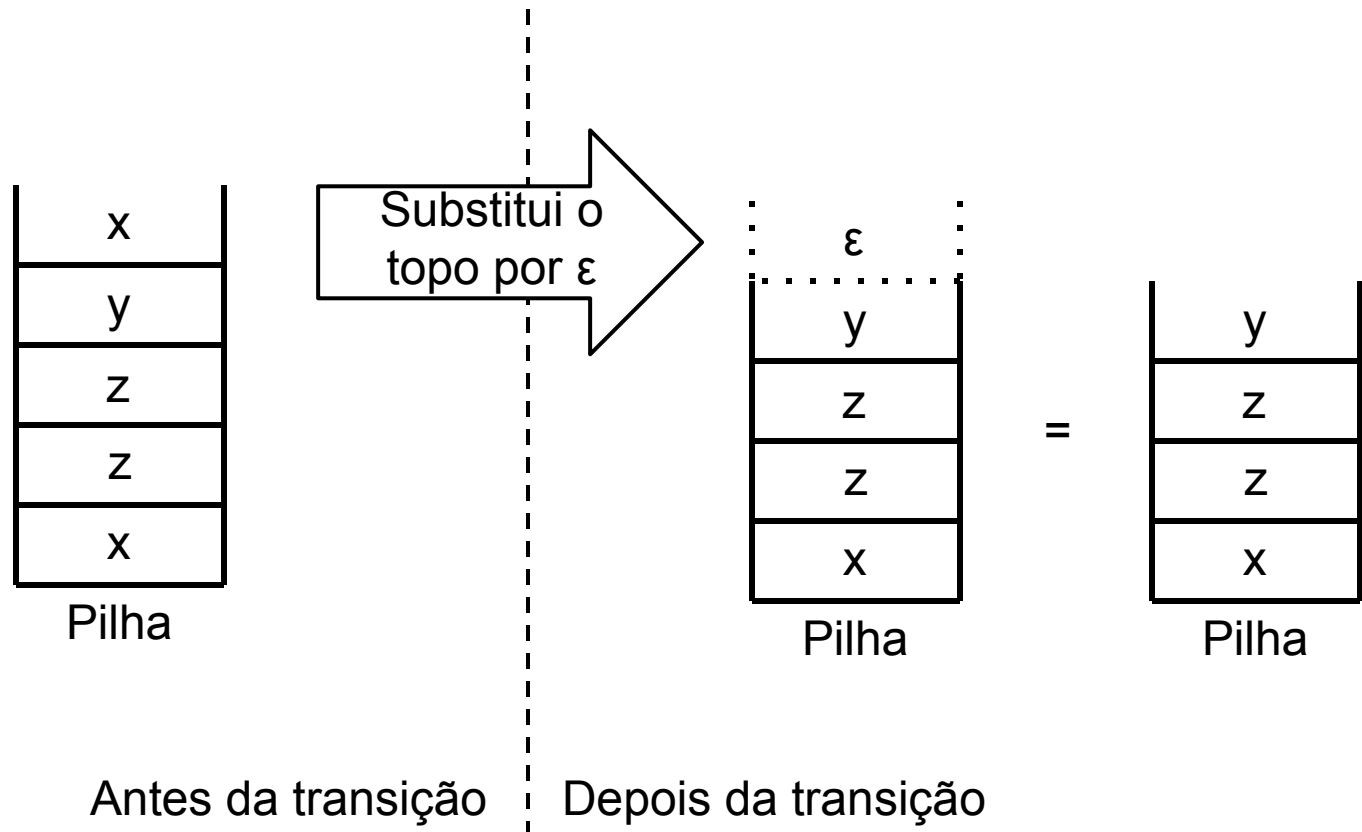
- O controle de estados finitos lê as entradas, um símbolo de cada vez
- O controle tem permissão para observar o símbolo no topo da pilha
  - Pode basear a transição:
    - em seu estado atual
    - no símbolo de entrada
    - no símbolo presente no topo da pilha
  - Opcionalmente, a entrada pode ser  $\epsilon$ 
    - Ou seja, podem haver transições “espontâneas”

# Autômatos com pilha

- Em uma transição, o autômato com pilha:
  - Consome da entrada o símbolo que utiliza na transição
    - Se for uma transição espontânea, nenhum símbolo de entrada é consumido
  - Vai para um novo estado, que pode ou não ser o mesmo estado anterior
  - Substitui o símbolo no topo da pilha por qualquer cadeia

# Autômatos com pilha

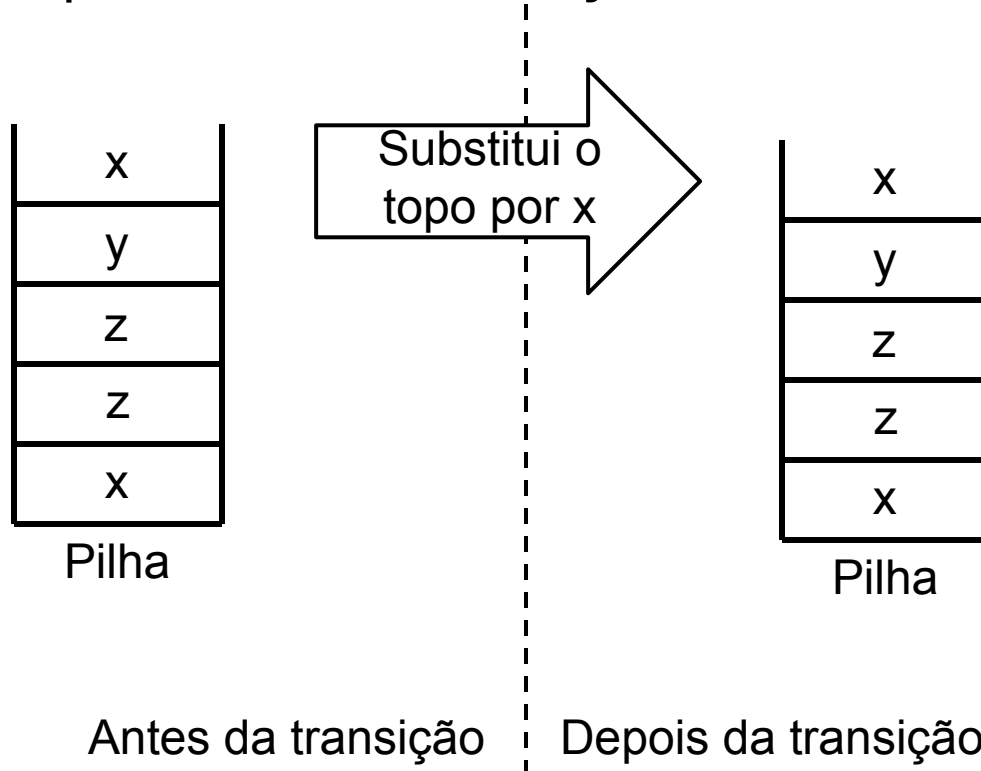
- Substituição de símbolo no topo da pilha
  - Se for  $\epsilon$ , equivale a uma extração (pop) da pilha





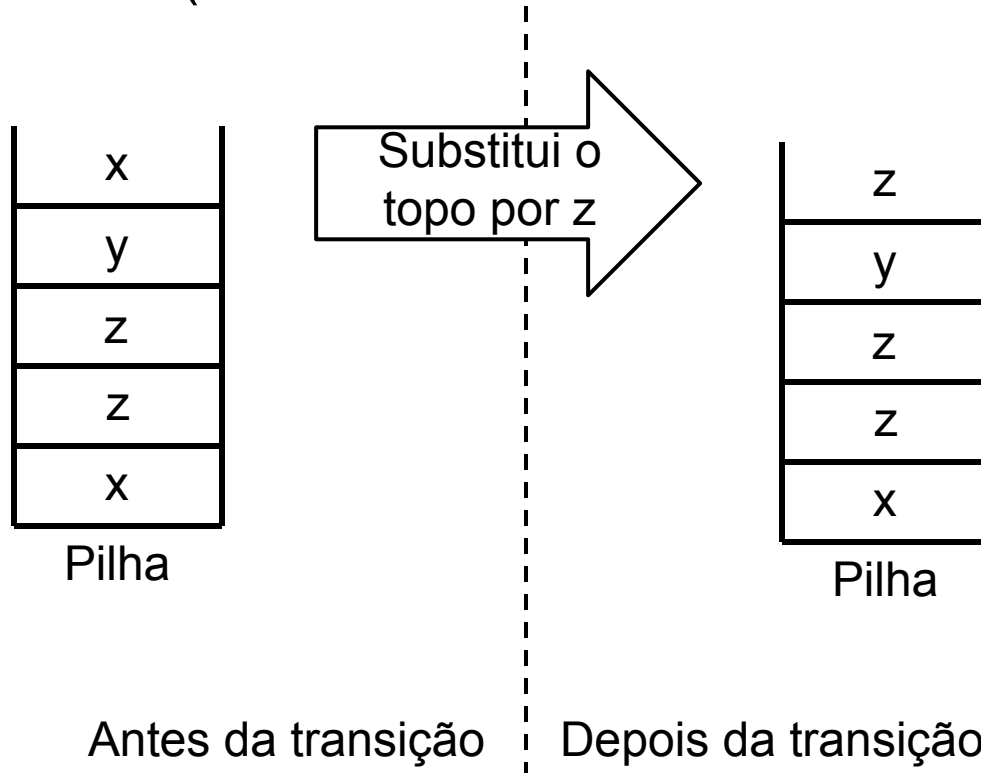
# Autômatos com pilha

- Substituição de símbolo no topo da pilha
  - Se for o mesmo que já estava, equivale a não mudar a pilha, apenas fazer a transição



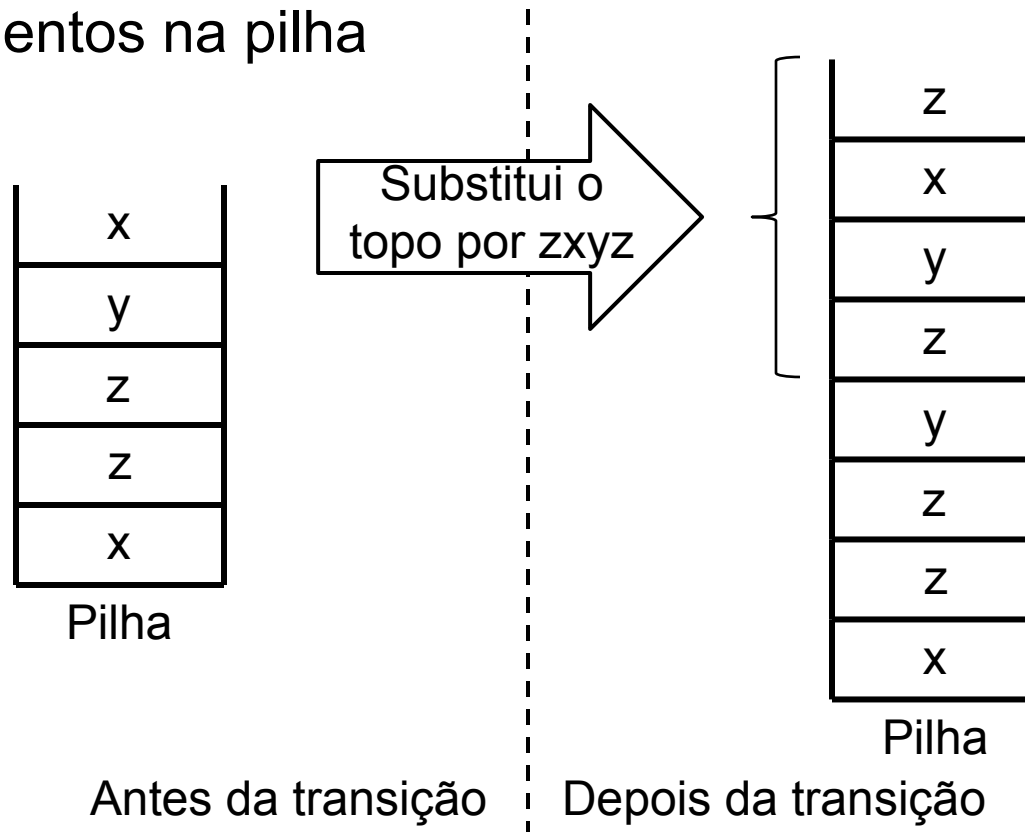
# Autômatos com pilha

- Substituição de símbolo no topo da pilha
  - Se for outro símbolo, altera o topo, mas não insere nem extrai nada (mantém o número de símbolos na pilha)



# Autômatos com pilha

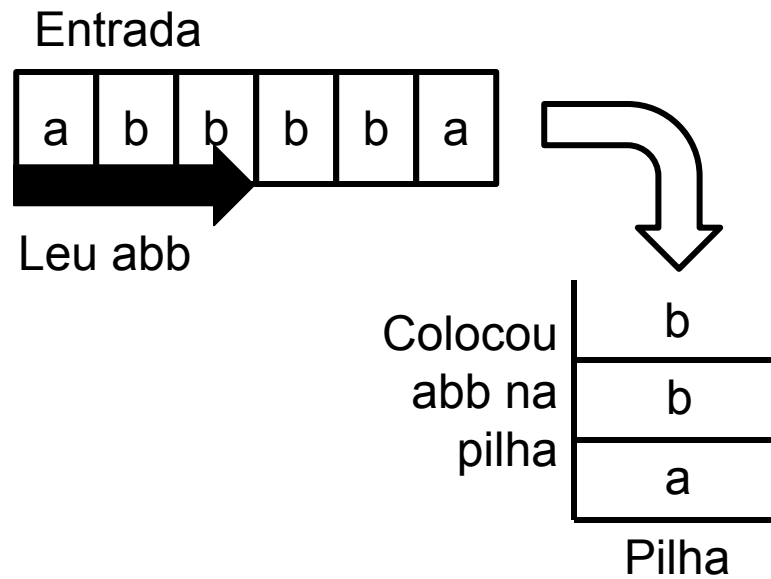
- Substituição de símbolo no topo da pilha
  - Se for uma cadeia com dois ou mais símbolos, insere elementos na pilha



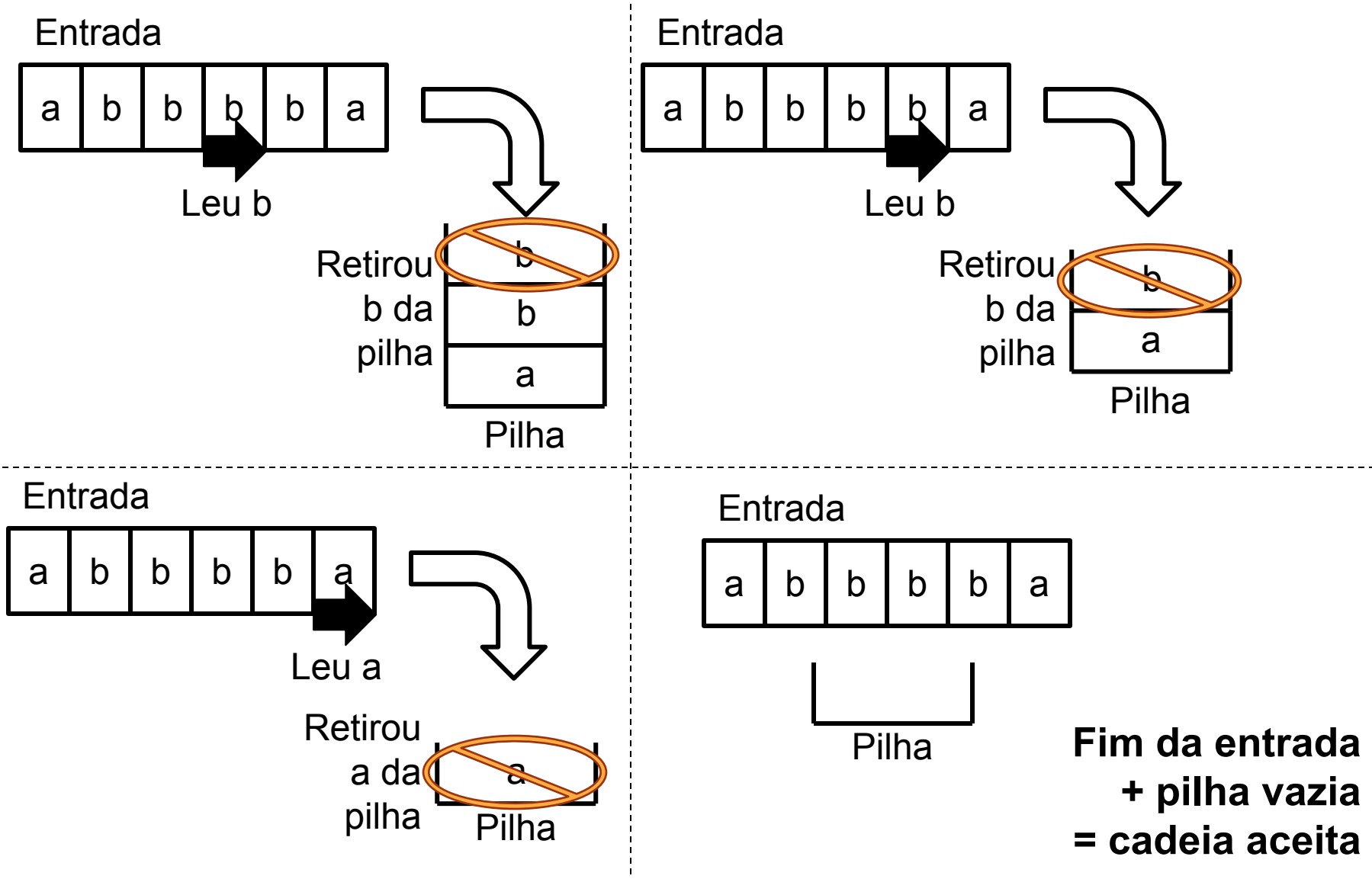
# Autômatos com pilha

- Ex:  $L = \{ww^R \mid w \text{ está em } \{0,1\}^*\}$ 
  - Três estados:  $q_0$ ,  $q_1$  e  $q_2$
  - $q_0$  = ainda não chegou no meio da entrada
  - $q_1$  = passou do meio da entrada
  - $q_2$  = chegou no fim da entrada
- Funcionamento:
  - Começa em  $q_0$
  - Estando em  $q_0$ , vai lendo a entrada e colocando uma cópia do símbolo lido no topo da pilha
  - No meio da pilha (usando o poder de oráculo do não-determinismo), muda para  $q_1$
  - Estando em  $q_1$ , vai lendo a entrada. Compara-se o símbolo lido com o símbolo no topo da pilha. Se for igual, substitui o topo da pilha por  $\epsilon$
  - No final da entrada, se a pilha estiver vazia (topo =  $\epsilon$ ), aceita a cadeia

# Autômatos com pilha



# Autômatos com pilha



# Autômatos com pilha

- Definição formal de um PDA (*Push-Down Automaton*)

$$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$$

- Onde:
  - $Q$  = Um conjunto finito de estados
  - $\Sigma$  = Um conjunto finito de símbolos de entrada
  - $\Gamma$  = Um alfabeto de pilha finito – conjunto de símbolos que temos permissão para inserir na pilha (pode incluir elementos de  $\Sigma$ )
  - $\delta$  = Função de transição – governa o comportamento do autômato
  - $q_0$  = Estado inicial
  - $Z_0$  = Símbolo de início – Inicialmente, a pilha do PDA consiste em uma instância desse símbolo e em nada mais
  - $F$  = Conjunto de estados de aceitação

# Autômatos com pilha

- Função de transição
  - $\delta : Q \times \Sigma \cup \{\varepsilon\} \times \Gamma \cup \{\varepsilon\} \rightarrow P(Q \times \Gamma \cup \{\varepsilon\})$
  - O argumento é uma tripla  $(q, a, X)$ , onde:
    - $q$  é um estado em  $Q$
    - $a$  é um símbolo de entrada em  $\Sigma$  ou  $a=\varepsilon$  (cadeia vazia)
    - $X$  é um símbolo da pilha, isto é, um elemento de  $\Gamma$
  - A saída de  $\delta$  é um conjunto finito de pares  $(p, \gamma)$ , onde:
    - $p$  é o novo estado
    - $\gamma$  é a cadeia de símbolos da pilha que substitui  $X$  no topo da pilha
      - Se  $\gamma = \varepsilon$ , a pilha é extraída
      - Se  $\gamma = X$ , a pilha fica inalterada
      - Se  $\gamma = YZ$ ,  $X$  é substituído por  $Z$  e  $Y$  é inserido na pilha



# Autômatos com pilha

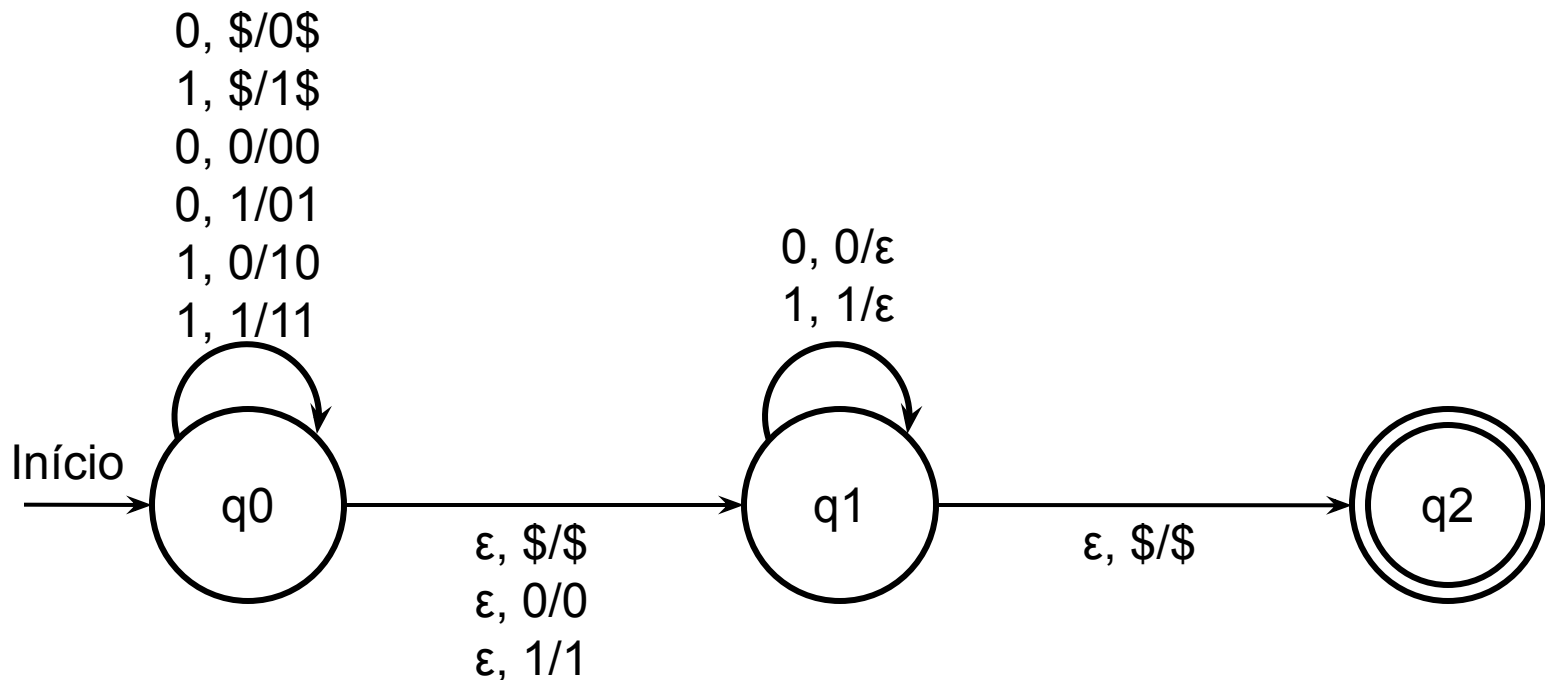
- Exemplo: Vamos projetar um PDA  $P$  para aceitar a linguagem  $L = \{ww^R \mid w \text{ está em } \{0,1\}^*\}$ :
- $P = (\{q_0, q_1, q_2\}, \{0,1\}, \{0,1,\$, \delta, q_0, \$, \{q_2\})$
- Obs:  $\$$  é um símbolo que fica no fundo da pilha, inicialmente, e será usado para marcar quando a pilha está vazia
- Onde  $\delta$  é definido pelas seguintes regras:
- Empilhando
  - $\delta(q_0, 0, \$) = \{(q_0, 0\$)\}$  e  $\delta(q_0, 1, \$) = \{(q_0, 1\$)\}$
  - $\delta(q_0, 0, 0) = \{(q_0, 00)\}$ ,  $\delta(q_0, 0, 1) = \{(q_0, 01)\}$ ,  $\delta(q_0, 1, 0) = \{(q_0, 10)\}$  e  $\delta(q_0, 1, 1) = \{(q_0, 11)\}$
- Adivinhando o meio da cadeia
  - $\delta(q_0, \epsilon, \$) = \{(q_1, \$)\}$ ,  $\delta(q_0, \epsilon, 0) = \{(q_1, 0)\}$  e  $\delta(q_0, \epsilon, 1) = \{(q_1, 1)\}$
- Desempilhando
  - $\delta(q_1, 0, 0) = \{(q_1, \epsilon)\}$  e  $\delta(q_1, 1, 1) = \{(q_1, \epsilon)\}$
- Checando a pilha vazia
  - $\delta(q_1, \epsilon, \$) = \{(q_2, \$)\}$

# Autômatos com pilha

- A lista de transições para um PDA nem sempre é fácil de acompanhar
- Uma forma melhor é um diagrama de transição para PDAs:
  - Os nós correspondem aos estados do PDA
  - Uma seta identificada por Início indica o estado inicial, e estados com círculos duplos são estados de aceitação
  - Os arcos correspondem a transições do PDA
    - Mas com algumas extensões, conforme a seguir

# Autômatos com pilha

- Um arco é identificado por  $a, X/\alpha$ 
  - $a$  = símbolo de entrada
  - $X$  = símbolo no topo da pilha
  - $\alpha$  = cadeia a substituir o topo da pilha



# Autômatos com pilha

- Outra forma é criar uma tabela tridimensional

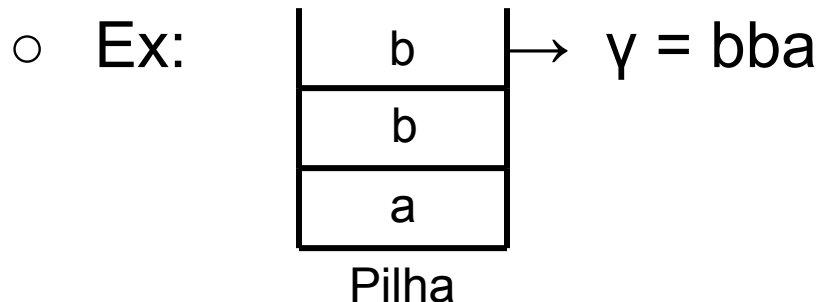
[illegible]

# Autômatos com pilha

- Para facilitar o acompanhamento da execução de um autômato, existe também o conceito de configuração (ou descrição) instantânea
  - Um texto que resume o estado da execução em um determinado momento, com as informações essenciais
    - Estado atual do PDA
    - Entrada a ser lida
    - Conteúdo da pilha

# Autômatos com pilha

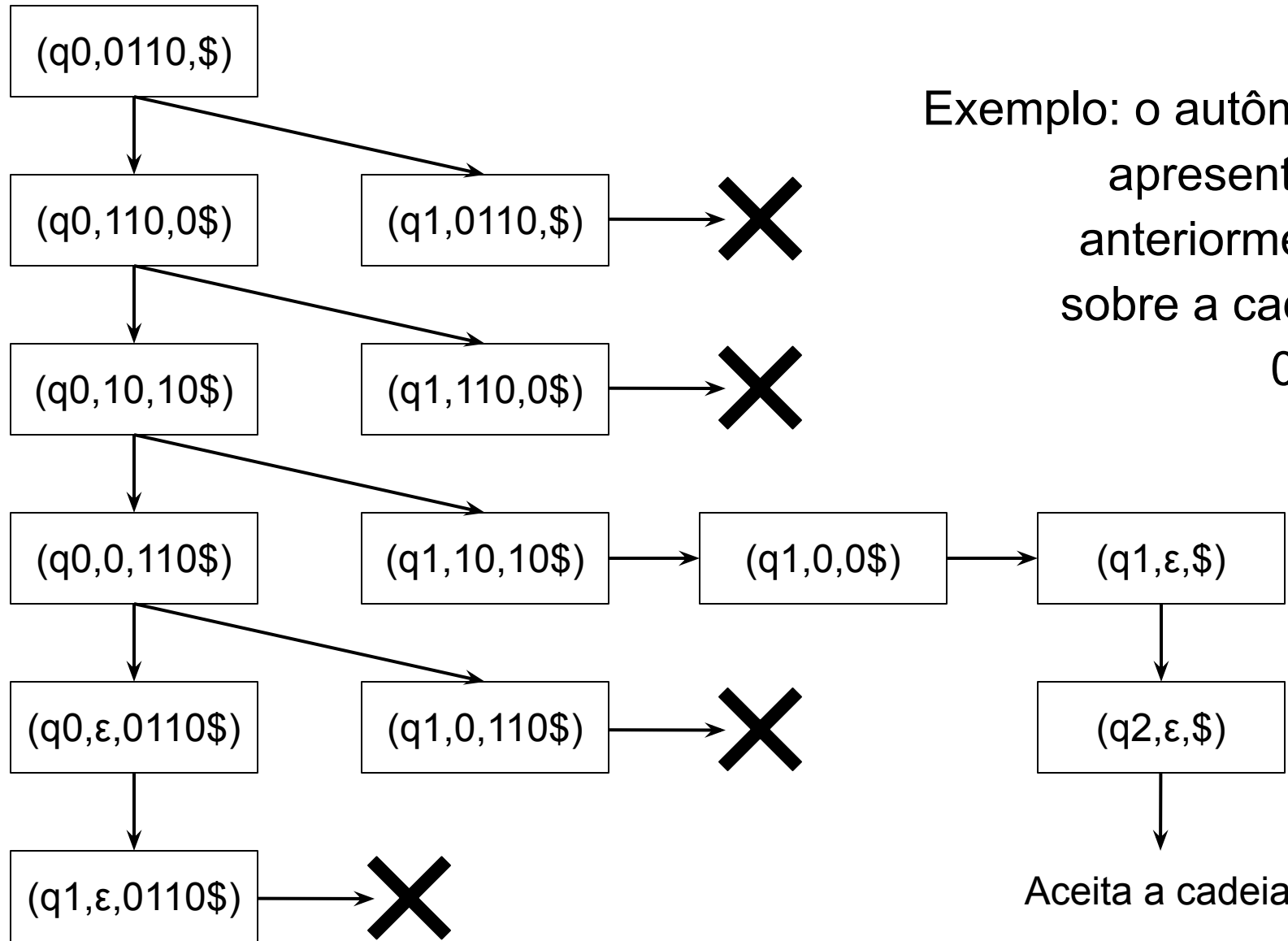
- A configuração instantânea (CI) de um PDA é representada por uma tripla  $(q, w, \gamma)$ , onde:
  - $q$  é o estado
  - $w$  é a parte restante da entrada
  - $\gamma$  é o conteúdo da pilha
- Convencionalmente, mostramos o topo da pilha na extremidade esquerda e a parte inferior na extremidade direita



# Autômatos com pilha

- Formalmente: um movimento genérico em um PDA é descrito pelas seguintes configurações instantâneas
  - $(q, aw, X\beta) \vdash (p, w, \alpha\beta)$
- Supondo que  $\delta(q, a, X)$  contém  $(p, \alpha)$ 
  - A notação  $\vdash^*$  indica uma sequência de movimentos

# Autômatos com pilha



Exemplo: o autômato  
apresentado  
anteriormente  
sobre a cadeia  
0110



# Autômatos com pilha

- Exercício: mostre as sequências de configurações instantâneas para a cadeia 1111

# Autômatos com pilha

- As linguagens de um PDA
- Existem duas abordagens para decidir se um PDA aceita ou não uma entrada
  - Aceitação pelo estado final
  - Aceitação por pilha vazia
- Ambas são equivalentes
  - Uma linguagem  $L$  tem um PDA que a aceita pelo estado final se e somente se  $L$  tem um PDA que a aceita por pilha vazia
  - Ou seja, é possível converter um PDA que aceita  $L$  por estado final em outro PDA que aceita  $L$  por pilha vazia
    - E vice-versa

# Autômatos com pilha

- Aceitação por estado final
  - Seja  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  um PDA
    - Então  $L(P)$ , a linguagem aceita por  $P$  pelo estado final, é:
      - $L(P) = \{w \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \alpha)\}$
      - Para algum estado  $q$  em  $F$  e qualquer cadeia de pilha  $\alpha$
- Ou seja, é o conjunto de todas as cadeias que o PDA pode processar, a partir da configuração instantânea inicial, consumindo todos os símbolos da cadeia, que chegam a um estado de aceitação
  - Não interessa o que “sobrar” na pilha

# Autômatos com pilha

- Aceitação por pilha vazia
  - Seja  $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$  um PDA
    - Então  $N(P)$ , a linguagem aceita por  $P$  por pilha vazia, é:
      - $N(P) = \{w \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon)\}$
      - Para qualquer estado  $q$  (não necessariamente em  $F$ )
- Ou seja, é o conjunto de cadeias que o PDA pode processar, a partir da configuração instantânea inicial, consumindo todos os símbolos da cadeia e deixando a pilha vazia no final
  - Não interessa em qual estado o autômato parou
  - Portanto, quando a aceitação é por pilha vazia, a descrição pode omitir o último elemento da tupla:
    - $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$

# Autômatos com pilha

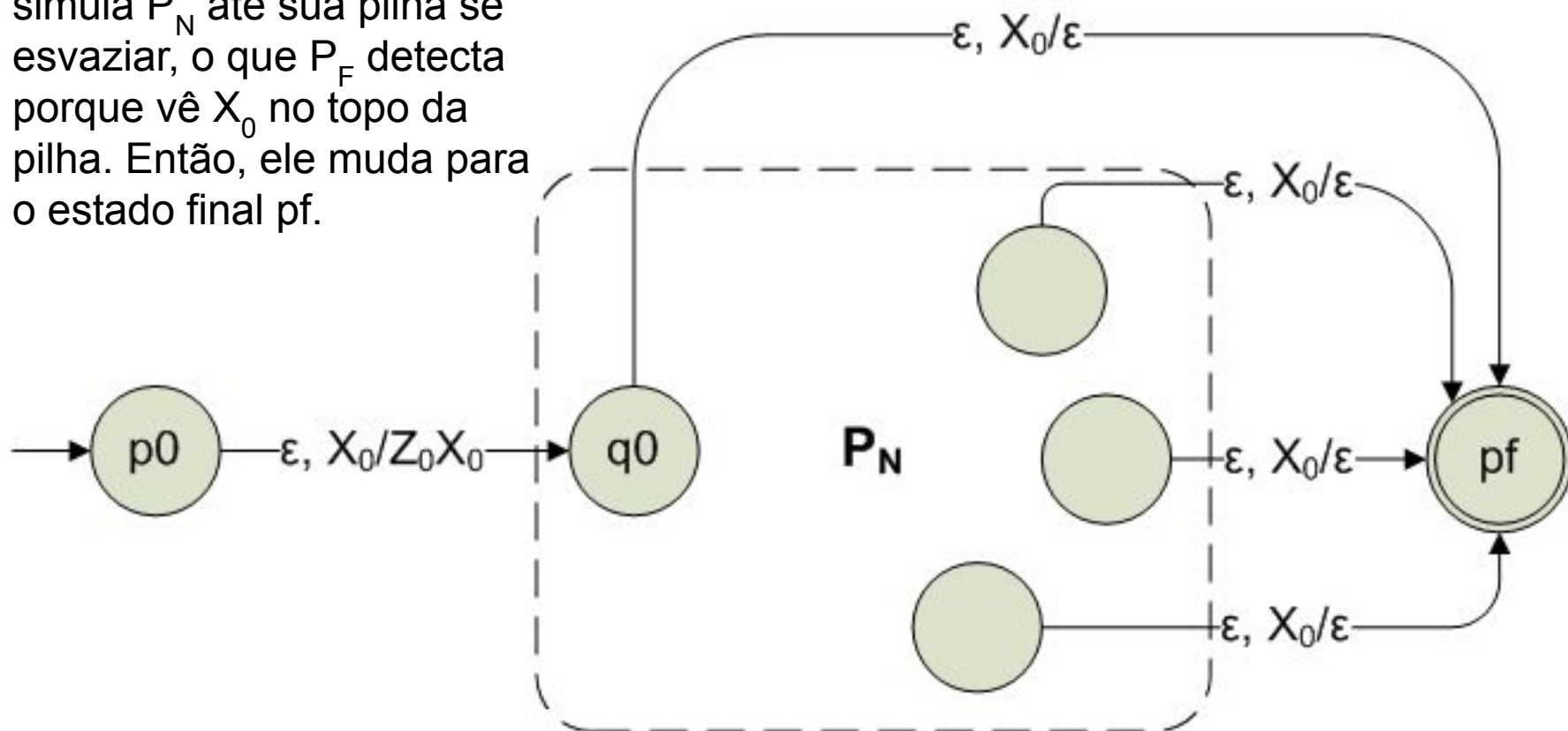
- Aceitação por pilha vazia = Aceitação por estado final
- Prova: por construção
  - Conversão de pilha vazia  $\rightarrow$  estado final
  - Conversão de estado final  $\rightarrow$  pilha vazia

# De pilha vazia ao estado final

- Dado um PDA  $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0, F)$  que aceita por pilha vazia:
- Criaremos um PDA  $P_F$  que aceita por estado final:
  - $P_F = (Q \cup \{p_0, p_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta_F, p_0, X_0, \{p_f\})$
  - Novo símbolo  $X_0$
  - Novo estado inicial  $p_0$
  - Novo estado final  $p_f$
  - Novas transições ( $\delta_F$ ) conforme a seguir

# De pilha vazia ao estado final

$P_F$  começa com  $X_0$ . O primeiro movimento é empilhar  $Z_0$ . Em seguida, simula  $P_N$  até sua pilha se esvaziar, o que  $P_F$  detecta porque vê  $X_0$  no topo da pilha. Então, ele muda para o estado final pf.



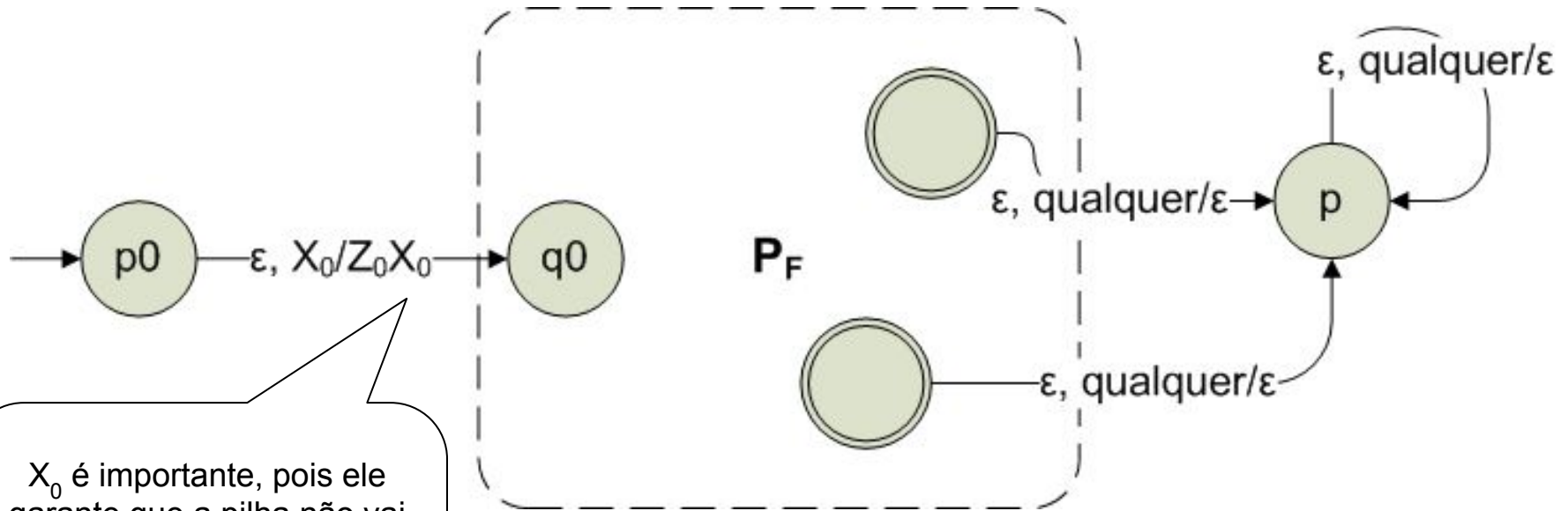
# De estado final para pilha vazia

- Dado um PDA  $P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$  que aceita por estado final:
- Criaremos um PDA  $P_N$  que aceita por pilha vazia:
  - $P_F = (Q \cup \{p_0, p\}, \Sigma, \Gamma \cup \{X_0\}, \delta_N, p_0, X_0)$
  - Novo símbolo  $X_0$
  - Novo estado inicial  $p_0$
  - Novo estado  $p$  (que representa a pilha vazia)
  - Novas transições ( $\delta_N$ ) conforme a seguir



# De estado final para pilha vazia

$P_N$  começa com  $X_0$ . Inicialmente, empilha  $Z_0$  e simula  $P_F$  até chegar a um estado de aceitação. Então, ele muda para um estado  $p$  que nada faz, a não ser esvaziar a pilha.



$X_0$  é importante, pois ele garante que a pilha não vai esvaziar acidentalmente, durante a simulação de  $P_F$  (o que pode acontecer, pois  $P_F$  aceita por estado final)

# Autômatos com pilha

- Se uma linguagem é  $L(P)$  ou  $N(P)$  para algum PDA, então  $L$  é livre de contexto
  - Prova: por construção
    - Conversão de CFG  $\rightarrow$  PDA
      - (Essencialmente, é o conteúdo da disciplina de compiladores, não vamos cobrir aqui)
    - Conversão de PDA  $\rightarrow$  CFG
      - (Não tem muita utilidade prática)

# Fim

Aula 17 - Autômatos com pilha