ACH2043
INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

Aula 12

Cap. 2.2 – Autômato com Pilha (cont.)

Profa. Ariane Machado Lima
ariane.machado@usp.br

Aula passada...

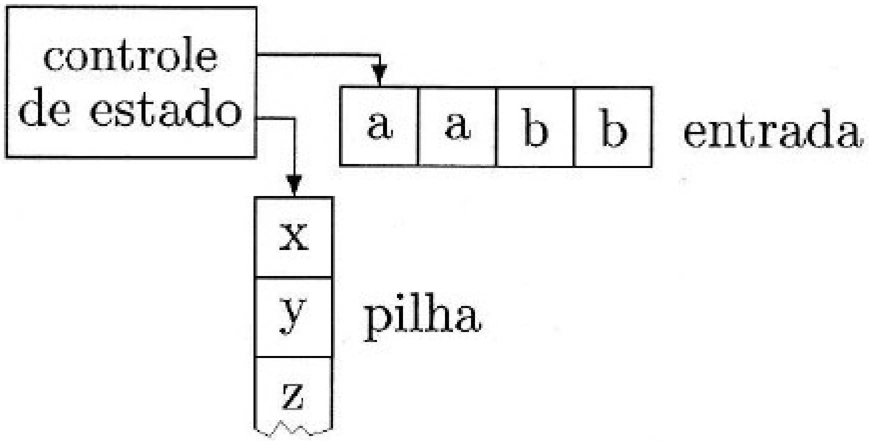
Algoritmo CYK para análise sintática

D = "On input $w = w_1 \cdots w_n$:

1. If $w = \varepsilon$ and $S \to \varepsilon$ is a rule, *accept*. [[handle $w = \varepsilon$ case]]
2. For $i = 1$ to n: [[examine each substring of length 1]]
3. For each variable A:
4. Test whether $A \to \mathtt{b}$ is a rule, where $\mathtt{b} = w_i$.
5. If so, place A in *table*(i, i).
6. For $l = 2$ to n: [[l is the length of the substring]]
7. For $i = 1$ to $n - l + 1$: [[i is the start position of the substring]]
8. Let $j = i + l - 1$, [[j is the end position of the substring]]
9. For $k = i$ to $j - 1$: [[k is the split position]]
10. For each rule $A \to BC$:
11. If *table*(i, k) contains B and *table*$(k + 1, j)$ contains C, put A in *table*(i, j).
12. If S is in *table*$(1, n)$, *accept*. Otherwise, *reject*."

Cap 2.2 – Autômato com pilha (AP)

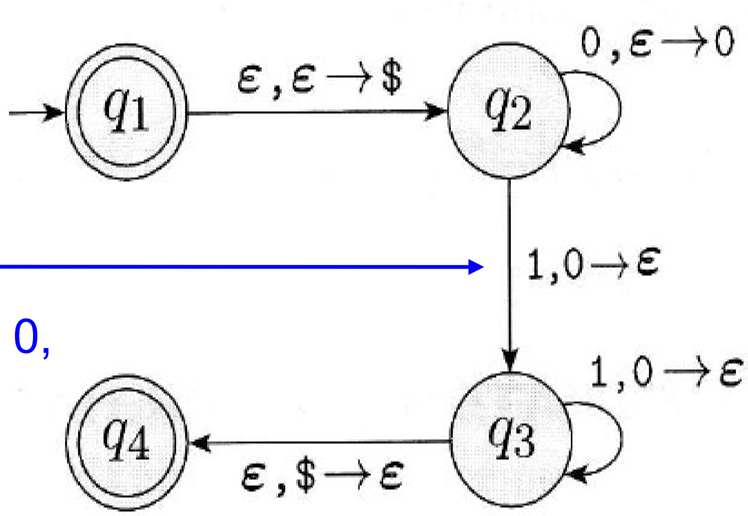
- Autômato finito com uma memória adicional (leitura e escrita DO TOPO da pilha)



- Lembram de B = $\{0^n1^n \mid n >= 0\}$?

Exemplo $B = \{0^n1^n \mid n >= 0\}$

Se estou em q2
e o próximo símbolo é 1
e o topo da pilha é 0, desempilho o 0,
empilho o épsilon e vou para q3



Aula de hoje...

Definição formal
Autômato com Pilha (AP) não determinístico

DEFINIÇÃO 2.13

Um ***autômato com pilha*** é uma 6-upla $(Q, \Sigma, \Gamma, \delta, q_0, F)$, onde Q, Σ, Γ e F são todos conjuntos finitos, e

1. Q é o conjunto de estados,
2. Σ é o alfabeto de entrada,
3. Γ é o alfabeto de pilha,
4. $\delta\colon Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow \mathcal{P}(Q \times \Gamma_\varepsilon)$ é a função de transição,
5. $q_0 \in Q$ é o estado inicial, e
6. $F \subseteq Q$ é o conjunto de estados de aceitação.

Exemplo

$\{0^n 1^n | \, n \geq 0\}$. Suponha que M_1 seja $(Q, \Sigma, \Gamma, \delta, q_1, F)$

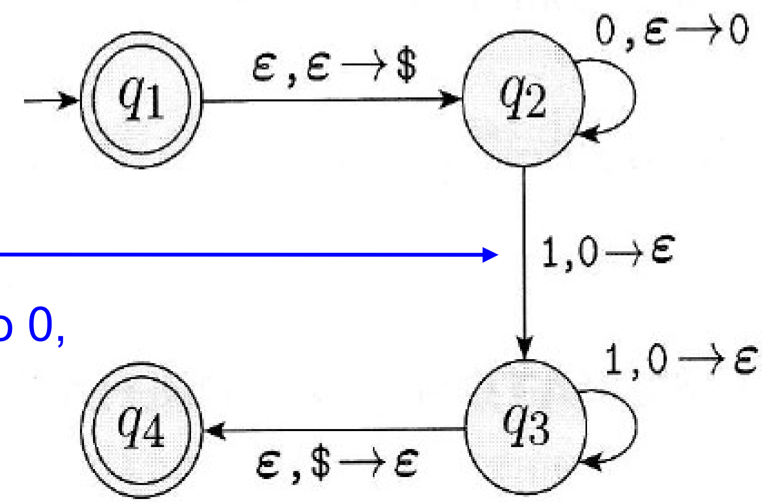
$Q = \{q_1, q_2, q_3, q_4\}$,

$\Sigma = \{0,1\}$,

$\Gamma = \{0, \$\}$,

$F = \{q_1, q_4\}$, e

Se estou em q2
e o próximo símbolo é 1
e o topo da pilha é 0, desempilho o 0,
empilho o épsilon e vou para q3



δ é dada pela tabela abaixo, na qual entradas em branco significam $\emptyset$.

Entrada:	0			1			ε		
Pilha:	0	\$	ε	0	\$	ε	0	\$	ε
q_1									$\{(q_2, \$)\}$
q_2			$\{(q_2, 0)\}$	$\{(q_3, \varepsilon)\}$					
q_3				$\{(q_3, \varepsilon)\}$				$\{(q_4, \varepsilon)\}$	
q_4									

Computação com um AP

Um autômato com pilha $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ computa da seguinte maneira. Ele aceita a entrada w se w puder ser escrita como $w = w_1 w_2 \cdots w_m$, onde cada $w_i \in \Sigma_\varepsilon$, e existem uma seqüência de estados $r_0, r_1, \ldots, r_m \in Q$ e cadeias $s_0, s_1, \ldots, s_m \in \Gamma^*$ que satisfazem as três condições a seguir. As cadeias s_i representam a seqüência de conteúdo da pilha que M tem no ramo de aceitação da computação.

1. $r_0 = q_0$ e $s_0 = \varepsilon$. Essa condição significa que M inicia apropriadamente, no estado inicial e com uma pilha vazia.
2. Para $i = 0, \ldots, m-1$, temos $(r_{i+1}, b) \in \delta(r_i, w_{i+1}, a)$, onde $s_i = at$ e $s_{i+1} = bt$ para algum $a, b \in \Gamma_\varepsilon$ e $t \in \Gamma^*$. Essa condição afirma que M se move apropriadamente, conforme o estado, a pilha e o próximo símbolo de entrada.
3. $r_m \in F$. Essa condição afirma que um estado de aceitação ocorre ao final da entrada.

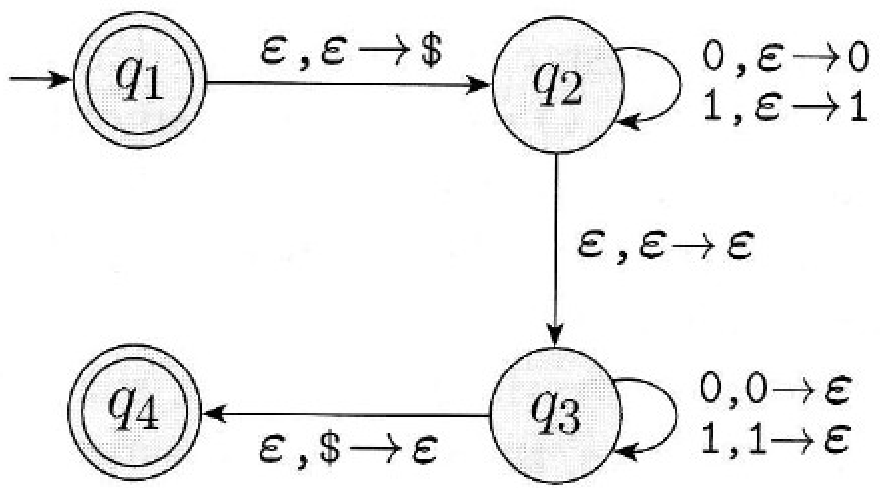
EXEMPLO 2.18

Nesse exemplo, damos um AP M_3 que reconhece a linguagem $\{ww^{\mathcal{R}} | \ w \in \{0,1\}^*\}$. Lembremo-nos de que $w^{\mathcal{R}}$ significa w escrita de trás para a frente.

EXEMPLO 2.18

Nesse exemplo, damos um AP M_3 que reconhece a linguagem $\{ww^{\mathcal{R}}|\ w \in \{0,1\}^*\}$. Lembremo-nos de que $w^{\mathcal{R}}$ significa w escrita de trás para a frente. Segue a descrição informal do AP.

> Comece empilhando os símbolos que são lidos. A cada ponto, adivinhe não-deterministicamente se o meio da cadeia foi atingido e, se tiver sido, passe a desempilhar um símbolo para cada símbolo lido, checando para garantir que eles sejam os mesmos. Se eles forem sempre os mesmos e a pilha esvaziar ao mesmo tempo em que a entrada terminar, aceite; caso contrário, rejeite.



Aqui não-determinismo é essencial!

EXEMPLO 2.16

$$\{a^i b^j c^k \mid i, j, k \geq 0 \text{ e } i = j \text{ ou } i = k\}$$

Empilho quando leio a's, e desempilho quando leio b's ou c's?

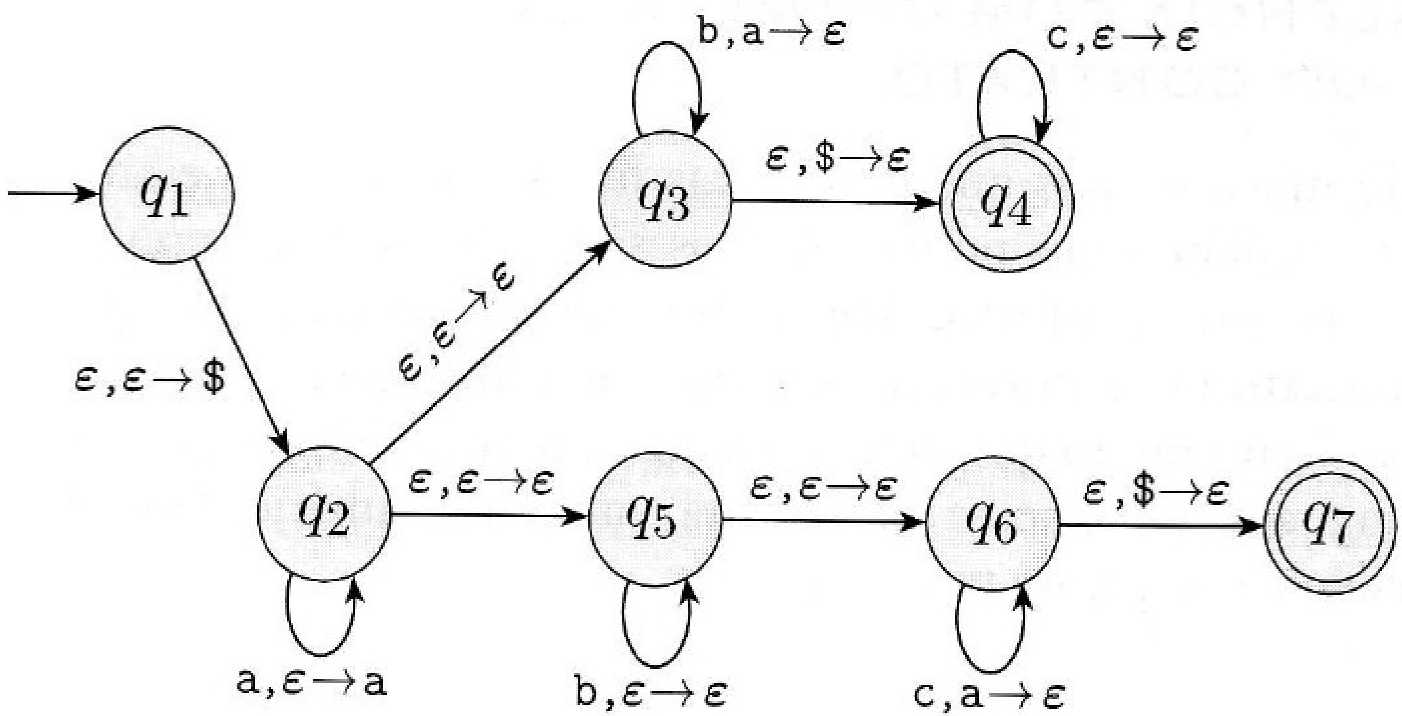
EXERCÍCIO !

EXEMPLO 2.16

$$\{a^i b^j c^k \mid i, j, k \geq 0 \text{ e } i = j \text{ ou } i = k\}$$

Empilho quando leio a's, e desempilho quando leio b's ou c's?

Aqui não-determinismo também é essencial!



Equivalência entre APN e GLC

TEOREMA 2.20

Uma linguagem é livre-do-contexto se e somente se algum autômato com pilha a reconhece.

<=>

NÃO DETERMINÍSTICO!!!

Equivalência entre APN e GLC

LEMA 2.21 ....

Se uma linguagem é livre-do-contexto, então algum autômato com pilha a reconhece. (=>)

Ideia da prova:

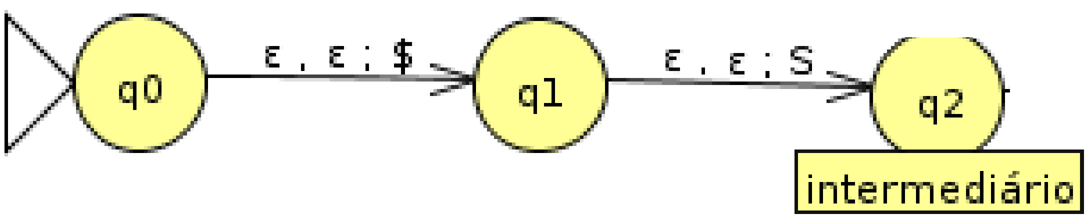
- Uma LLC é gerada por uma GLC
- Mostrar como converter uma GLC em um APN equivalente

Conversão GLC em APN (ideia)

- Uma gramática aceita uma cadeia w se, começando pela variável inicial, chega-se a uma cadeia apenas de símbolos terminais (w) após uma sequência de derivações diretas (substituições de variáveis).
- Um autômato aceita uma cadeia w se, começando pelo estado inicial, chega-se ao estado final após uma sequência de mudança de estados (transições)
- Simular cada substituição (da GLC) por uma transição (do APN)

Conversão GLC em APN (ideia)

- O APN empilha “$” (estado q0), e depois empilha a variável inicial na pilha (na transição do estado q1 para o estado intermediário)

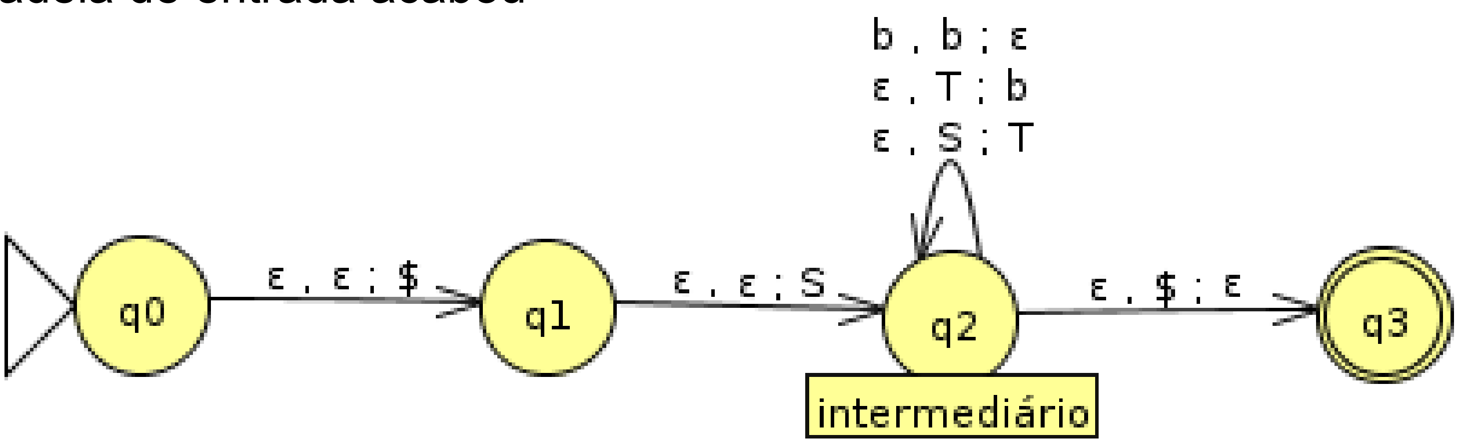


Conversão GLC em APN (ideia)

- O APN empilha “$” (estado q0), e depois empilha a variável inicial na pilha (na transição do estado q1 para o estado intermediário)
- O estado intermediário possui transições para ele mesmo, em cada uma fazendo uma substituição (derivação) na cadeia que está na pilha ou desempilhando um símbolo do alfabeto se este bater com o próximo símbolo de entrada
- O APN vai para o estado final quando não há mais substituições a serem feitas (pilha tem “$”) e a cadeia de entrada acabou
- Ex:

  $S \rightarrow T$

  $T \rightarrow b$



Conversão GLC em APN (ideia)

- O APN empilha “$” (estado q0), e depois empilha a variável inicial na pilha (na transição do estado q1 para o estado intermediário)
- O estado intermediário possui transições para ele mesmo, em cada uma fazendo uma substituição (derivação) na cadeia que está na pilha ou desempilhando um símbolo do alfabeto se este bater com o próximo símbolo de entrada
- O APN vai para o estado final quando não há mais substituições a serem feitas (pilha tem “$”) e a cadeia de entrada acabou
- Ex2:

  $S \rightarrow T$

  $T \rightarrow b \mid a$

Conversão GLC em APN (ideia)

- O APN empilha "$" (estado q0), e depois empilha a variável inicial na pilha (na transição do estado q1 para o estado intermediário)
- O estado intermediário possui transições para ele mesmo, em cada uma fazendo uma substituição (derivação) na cadeia que está na pilha ou desempilhando um símbolo do alfabeto se este bater com o próximo símbolo de entrada
- O APN vai para o estado final quando não há mais substituições a serem feitas (pilha tem "$") e a cadeia de entrada acabou
- Ex2:

  $S \rightarrow T$

  $T \rightarrow b \mid a$

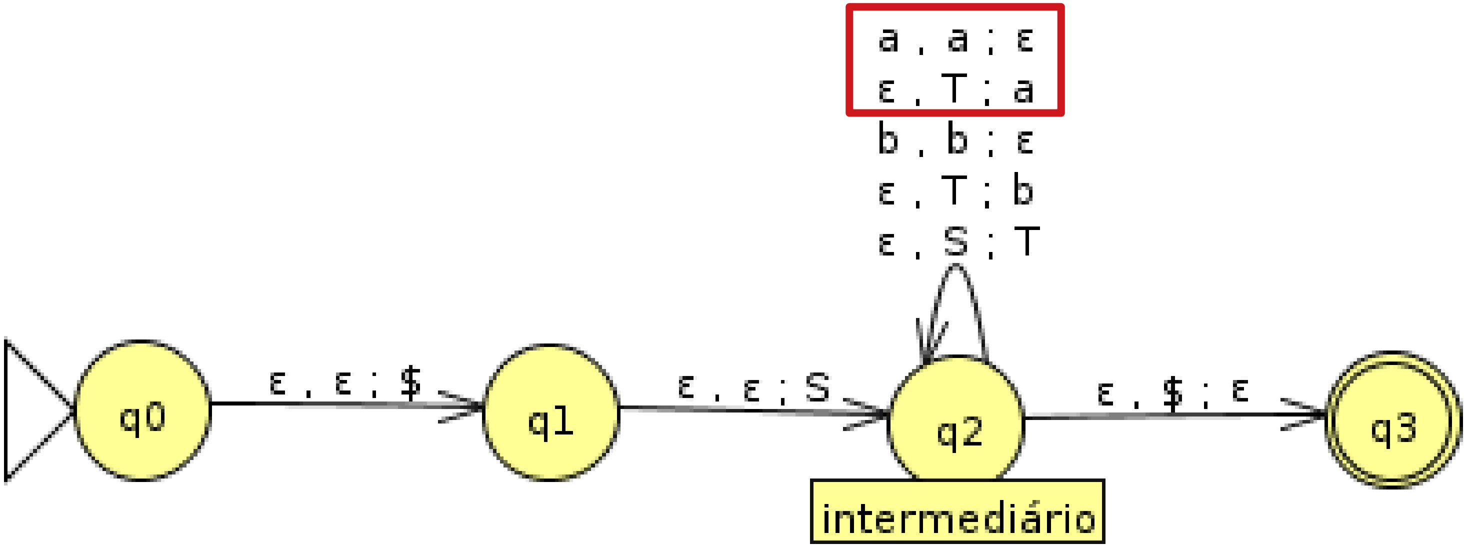
Problemas a serem resolvidos:

O que fazer quando há várias opções de substituição?

Conversão GLC em APN (ideia)

- Problema 1: O que fazer quando há várias opções de substituições?
  - Aproveitar o não determinismo

$S \to T$
$T \to b \mid a$



Conversão GLC em APN (ideia)

- O APN empilha “$” (estado q0), e depois empilha a variável inicial na pilha (na transição do estado q1 para o estado intermediário)
- O estado intermediário possui transições para ele mesmo, em cada uma fazendo uma substituição (derivação) na cadeia que está na pilha ou desempilhando um símbolo do alfabeto se este bater com o próximo símbolo de entrada
- O APN vai para o estado final quando não há mais substituições a serem feitas (pilha tem “$”) e a cadeia de entrada acabou
- Ex3:

$$S \to \mathtt{a}T\mathtt{b} \mid \mathtt{b}$$
$$T \to T\mathtt{a} \mid \varepsilon$$

Conversão GLC em APN (ideia)

- O APN empilha “$” (estado q0), e depois empilha a variável inicial na pilha (na transição do estado q1 para o estado intermediário)
- O estado intermediário possui transições para ele mesmo, em cada uma fazendo uma substituição (derivação) na cadeia que está na pilha ou desempilhando um símbolo do alfabeto se este bater com o próximo símbolo de entrada
- O APN vai para o estado final quando não há mais substituições a serem feitas (pilha tem “$”) e a cadeia de entrada acabou
- Ex3:

$$S \to \mathtt{a}T\mathtt{b} \mid \mathtt{b}$$
$$T \to T\mathtt{a} \mid \varepsilon$$

Problemas a serem resolvidos:

- O que fazer quando há várias opções de substituição?

- Como empilhar uma cadeia (e não apenas um símbolo)?

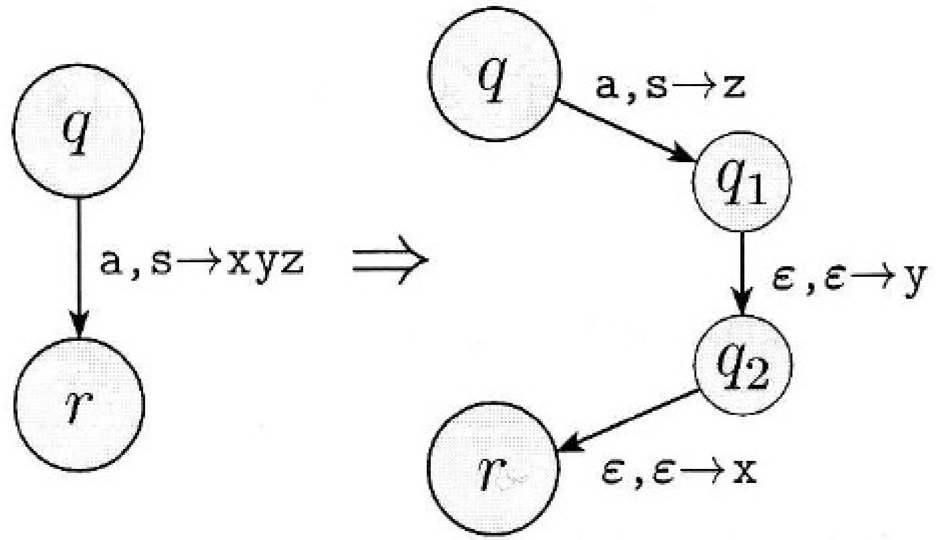
- Se só podemos ler o topo da pilha, o que fazer quando a primeira variável da forma sentencial não estiver no topo da pilha?

Conversão GLC em APN (ideia)

- Problema 2: Como empilhar uma cadeia, e não simplesmente um símbolo?

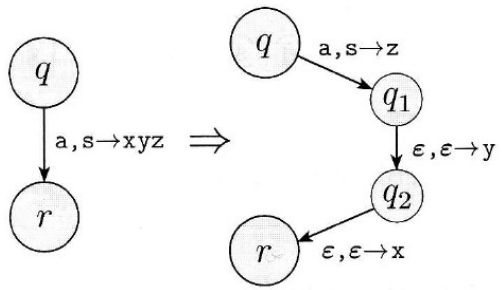
Conversão GLC em APN (ideia)

- Problema 2: Como empilhar uma cadeia, e não simplesmente um símbolo?



Conversão GLC em APN (ideia)

Problema 2: Como empilhar uma cadeia, e não simplesmente um símbolo?



Sejam q e r estados do AP e suponha que a esteja em Σ_ε e s em Γ_ε. Digamos que queiramos que o AP vá de q para r quando ele lê a e desempilha s. Além do mais, queremos empilhar a cadeia inteira $u = u_1 \cdots u_l$ ao mesmo tempo. Podemos implementar essa ação introduzindo novos estados $q_1, \ldots, q_{l-1}$ e montando a tabela de transição da seguinte maneira

$$
\begin{aligned}
&\delta(q, a, s) \text{ deve conter } (q_1, u_l),\\
&\delta(q_1, \varepsilon, \varepsilon) = \{(q_2, u_{l-1})\},\\
&\delta(q_2, \varepsilon, \varepsilon) = \{(q_3, u_{l-2})\},\\
&\qquad\vdots\\
&\delta(q_{l-1}, \varepsilon, \varepsilon) = \{(r, u_1)\}.
\end{aligned}
$$

Conversão GLC em APN (ideia)

- Problema 3: Se só podemos ler o topo da pilha, o que fazer quando a primeira variável da forma sentencial não estiver no topo da pilha?

Conversão GLC em APN (ideia)

- Problema 3: Se só podemos ler o topo da pilha, o que fazer quando a primeira variável da forma sentencial não estiver no topo da pilha?
  - Sempre faremos a derivação mais à esquerda
  - Se o começo da forma sentencial contiver terminais, desempilho esses símbolos “casando-os” com a entrada (por meio de transições).

Conversão GLC em APN (ideia)

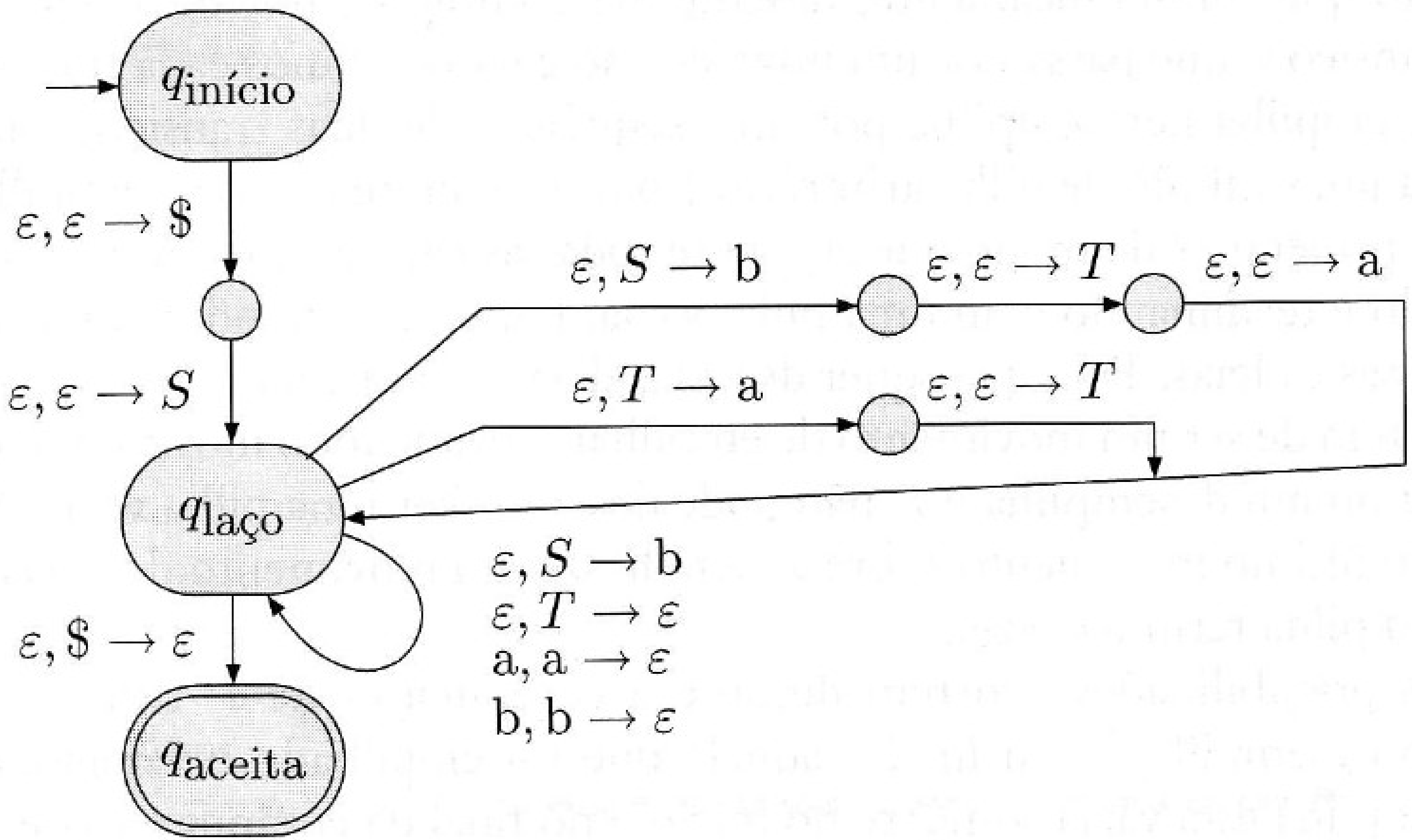
- Exemplo

$$S \to \mathtt{a}T\mathtt{b} \mid \mathtt{b}$$
$$T \to T\mathtt{a} \mid \varepsilon$$

Conversão GLC em APN (ideia)

- Exemplo

$$S \rightarrow \mathtt{a}T\mathtt{b} \mid \mathtt{b}$$
$$T \rightarrow T\mathtt{a} \mid \varepsilon$$

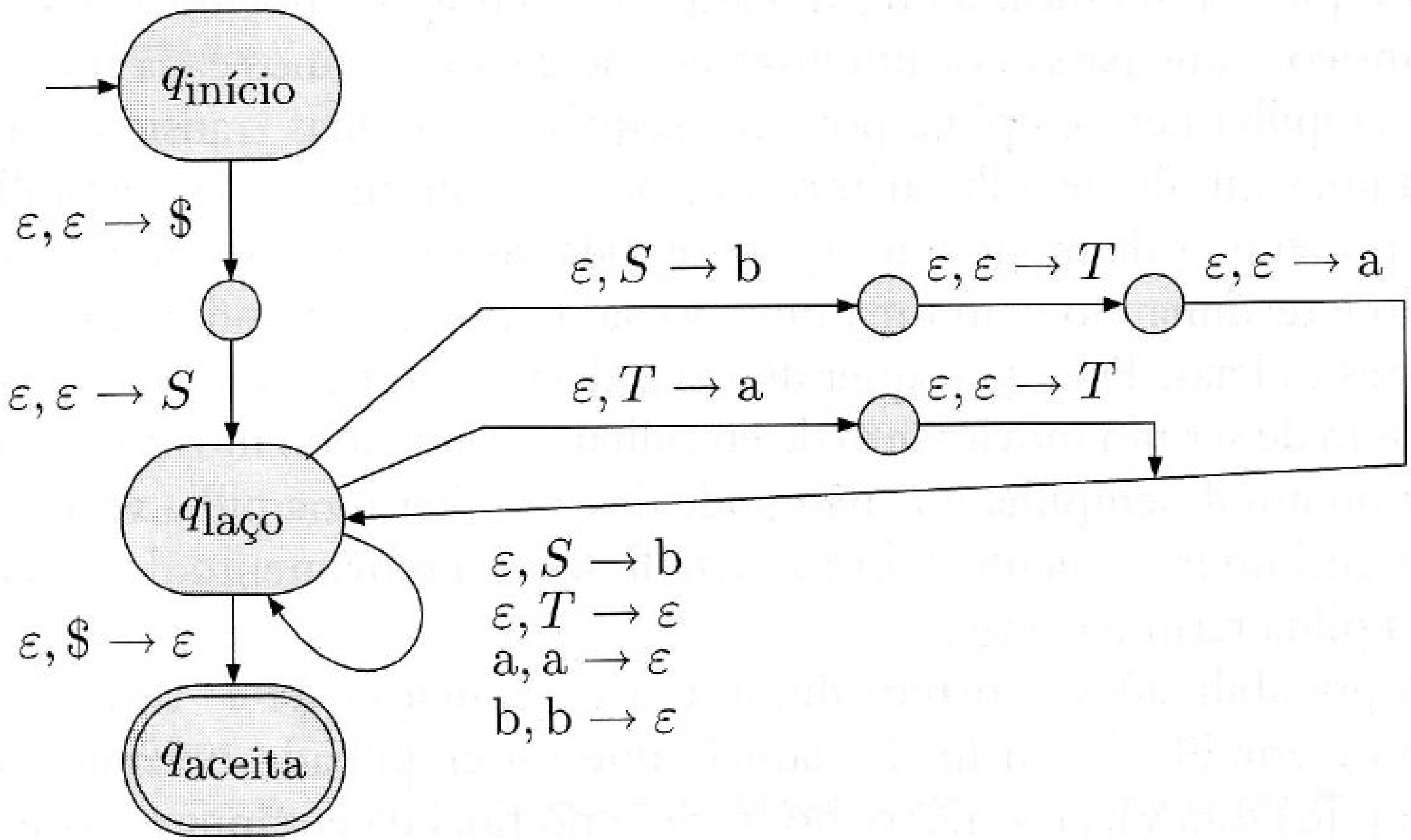


Conversão GLC em APN (ideia)

- Exemplo

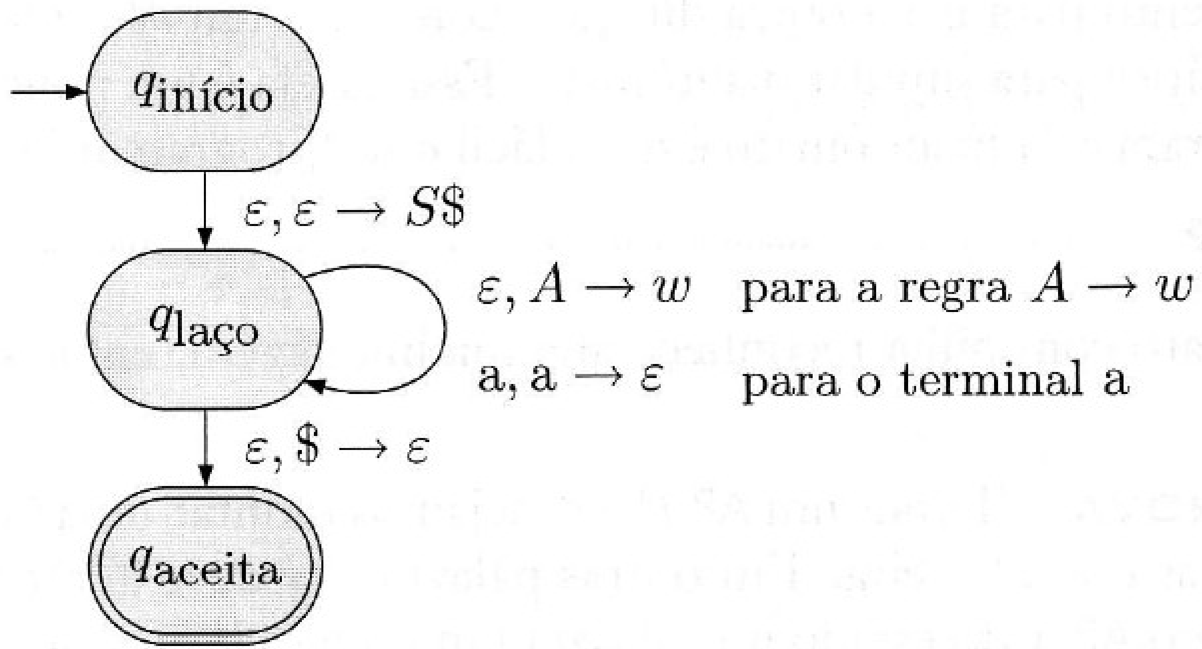
$$S \to \mathtt{a}T\mathtt{b} \mid \mathtt{b}$$
$$T \to T\mathtt{a} \mid \varepsilon$$

Como seria o reconhecimento da cadeia aaab?



Conversão GLC em APN (ideia)

- Caso Geral: (notação simplificada, representando o empilhamento de uma sequência de símbolos todos de uma vez)



Equivalência entre APN e GLC

TEOREMA 2.20

Uma linguagem é livre-do-contexto se e somente se algum autômato com pilha a reconhece.

PROVAMOS ESTE! (=>)

LEMA 2.21

Se uma linguagem é livre-do-contexto, então algum autômato com pilha a reconhece.

LEMA 2.27

Se um autômato com pilha reconhece alguma linguagem, então ela é livre-do-contexto.

Equivalência entre APN e GLC

TEOREMA 2.20 ....

Uma linguagem é livre-do-contexto se e somente se algum autômato com pilha a reconhece.

LEMA 2.21 ....

Se uma linguagem é livre-do-contexto, então algum autômato com pilha a reconhece.

LEMA 2.27 ....

Se um autômato com pilha reconhece alguma linguagem, então ela é livre-do-contexto.

FALTA PROVAR ESTE… (<=)