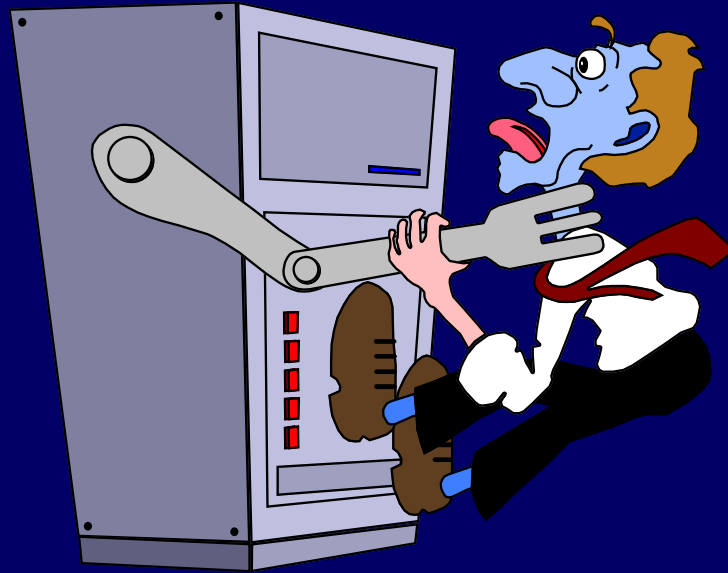


# Máquinas de Turing 2



Uma Máquina de Turing Exemplo

Máquinas de Turing com Fita Limitada ou Autômato  
Limitado Linearmente - ALL

(Linear Bounded Automata - LBA)

e sua relação com Linguagens Sensíveis ao Contexto

Exercícios

# Uma MT para uma Linguagem Simples

$$L = \{ a^n b^n c^n \mid n \geq 0 \}$$

Exemplos:

*Pertence à L:*

aaabbbccc

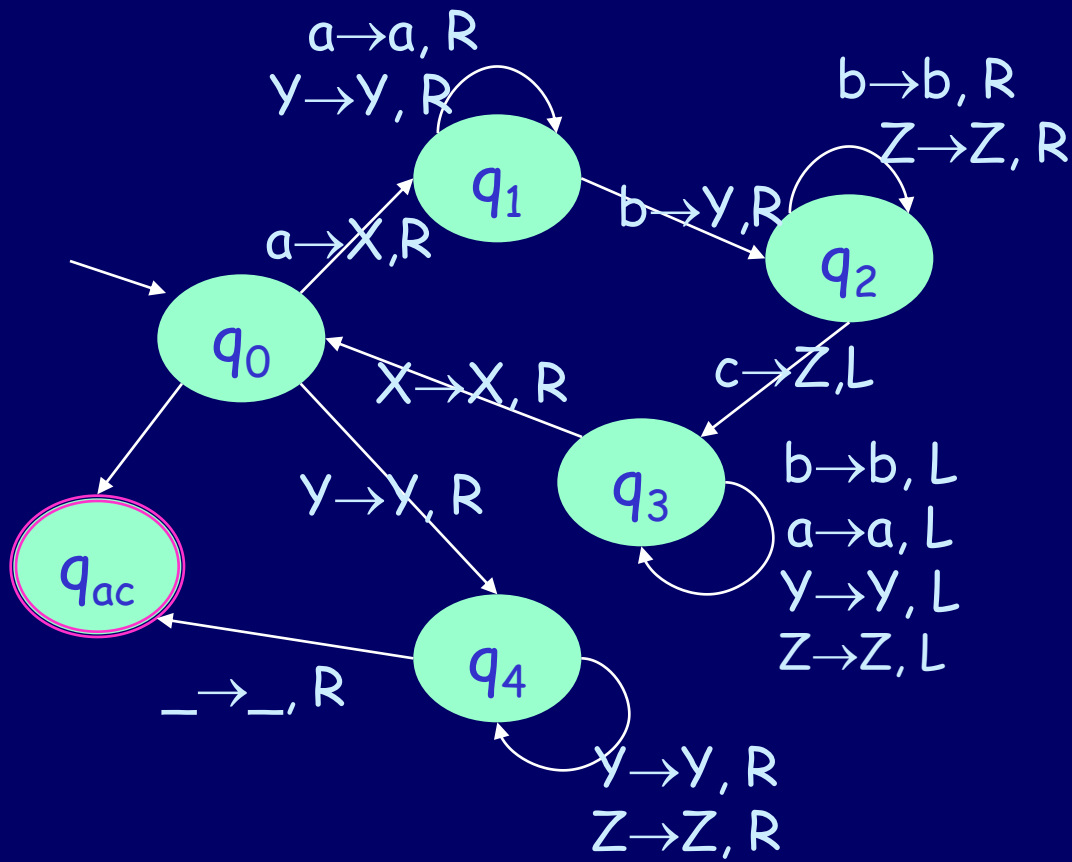
*Não Pertence à L:*

aaabbbcccc

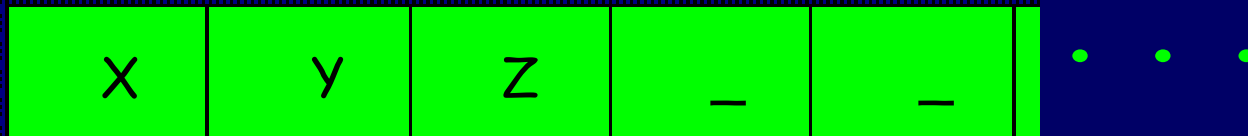
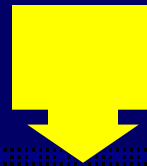
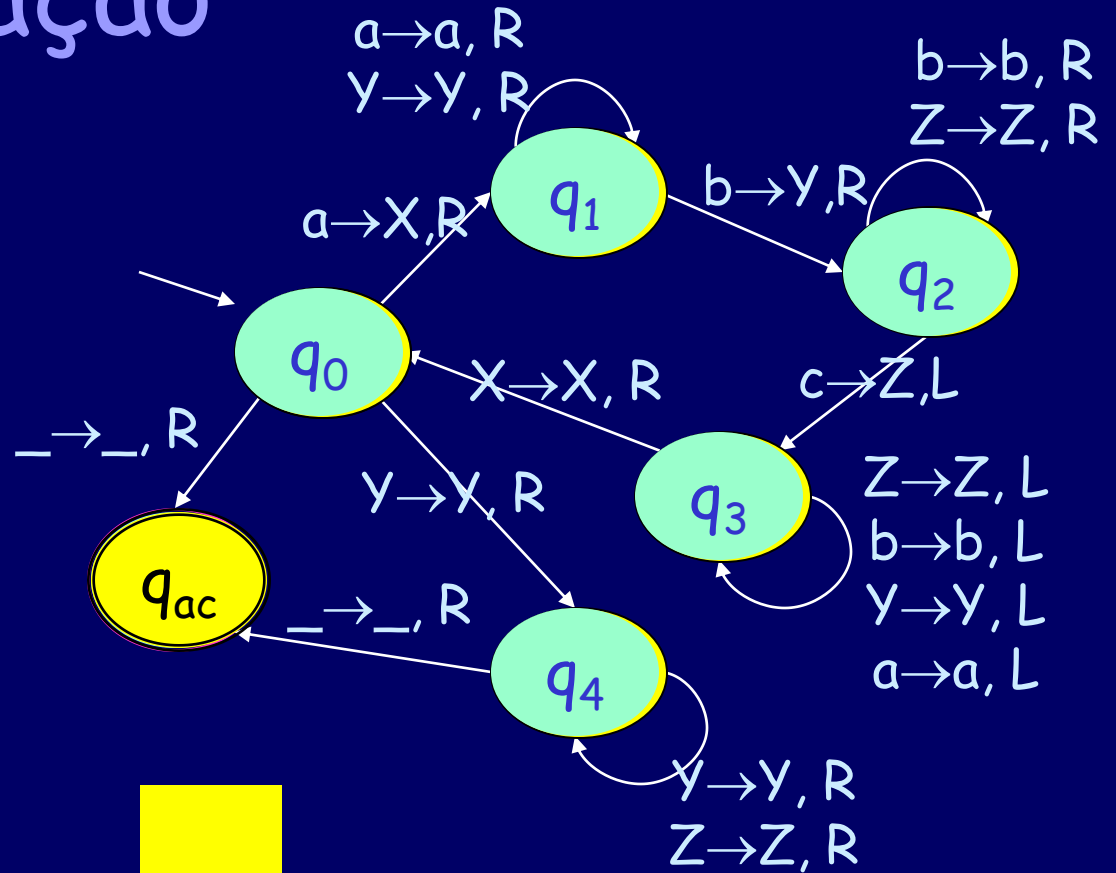
# A Máquina de Turing

1.  $Q = \{q_0, q_1, q_2, q_3, q_4, q_{\text{accept}}, q_{\text{reject}}\}$
2.  $\Sigma = \{a, b, c\}$
3.  $\Gamma = \{a, b, c, \_, X, Y, Z\}$
4.  $\delta$  a seguir.
5.  $q_0$  - o estado inicial.
6.  $q_{\text{accept}} \in Q$  - o estado de aceitação.
7.  $q_{\text{reject}} \in Q$  - o estado de rejeição.





# Demonstração



# Gramáticas Sensíveis ao Contexto

- $G = (V, \Sigma, R, S)$
- $V$  ou  $VN$  é o vocabulário não terminal
- $\Sigma$  ou  $VT$  é o vocabulário terminal
- $R$  são as regras da forma  
$$\alpha A \beta \rightarrow \alpha \gamma \beta$$

( $A$  é trocado por  $\gamma$  no contexto de  $\alpha$  e  $\beta$ )  
onde  $A \in V, \alpha, \beta \in (V \cup \Sigma)^*, e$   
 $\gamma \in (V \cup \Sigma)^+$   
mais a regra  $S \rightarrow \lambda$  se  $S$  não está do lado direito de qualquer regra.

- $| \alpha A \beta | \leq | \alpha \gamma \beta |$
- GSC são chamadas **não decrescentes** porque nenhuma regra diminui o tamanho da cadeia sendo gerada.
- Por exemplo:  
 $S \rightarrow aSBc \rightarrow aaSBcBc \rightarrow aaabcBcBc \rightarrow$   
 $aaabBcBcc \rightarrow aaabbBccc \rightarrow aaabbbccc$

# Exemplo clássico

- Considere a GSC:

$$S \rightarrow aSBc$$

$$S \rightarrow abc$$

$$cB \rightarrow Bc$$

$$bB \rightarrow bb$$

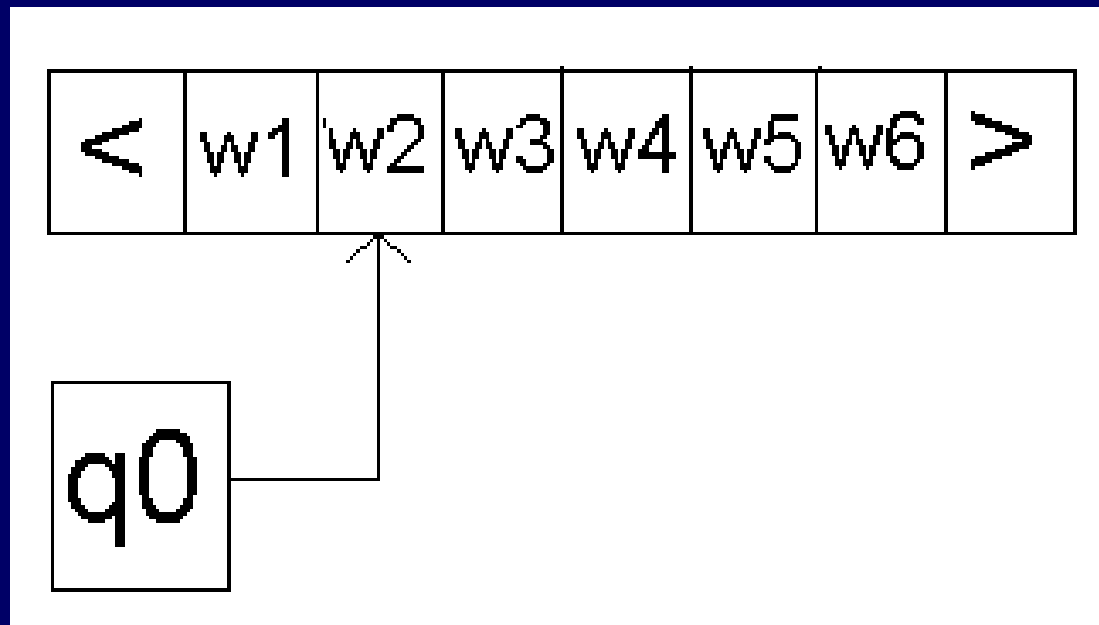
- A linguagem gerada é  $L(G) = \{a^n b^n c^n \mid n \geq 1\}$



# Máquinas de Turing com Fita Limitada

- Uma MT que tem o tamanho de sua fita limitado ao tamanho de sua cadeia de entrada é chamada *linear-bounded automaton* (LBA) ou **automato limitado linearmente (ALL)**
- Um LBA é uma MT **não determinística**:  
 $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$  exceto que:
  1. Existem dois símbolos **extra fita**  $\langle$  e  $\rangle$ , que não são elementos de  $\Gamma$ .
  2. A máquina começa na configuração  $(q_0, \langle x \rangle, 0)$ , com a cabeça apontando o símbolo  $\langle$  na célula 0. O símbolo  $\rangle$  está na célula imediatamente a direita da cadeia de entrada  $x$ .
  3. A MT não pode trocar  $\langle$  ou  $\rangle$  com qualquer outro símbolo, nem mover a cabeça da fita a esq de  $\langle$  ou a direita de  $\rangle$ .

# Automato limitado linearmente



$$L = \{a^n b^n c^n \mid n \geq 0\}$$

$$Q = \{s, t, u, v, w, \text{accept}\} \quad \Sigma = \{a, b, c\}$$

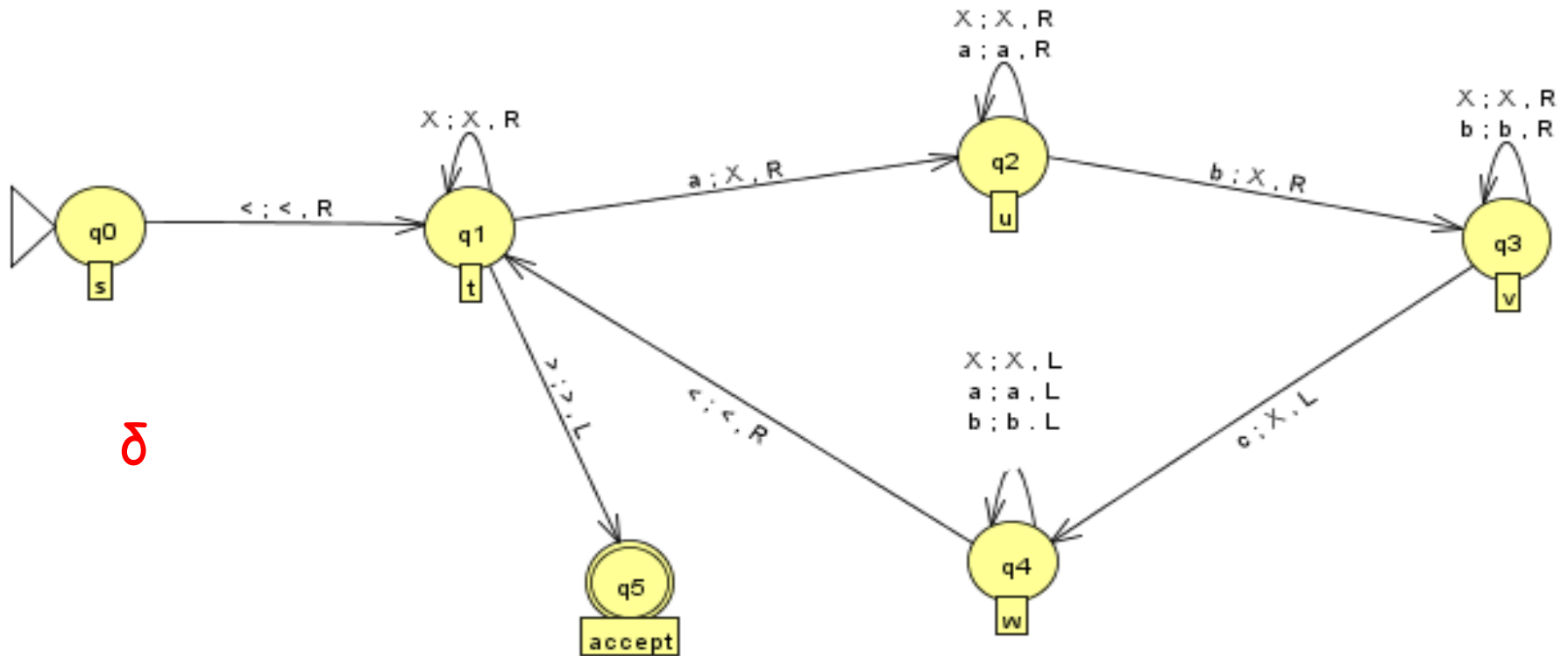
$$\Gamma = \{a, b, c, x\} \quad q_0 = s$$

$\delta =$

$$\begin{aligned} &\{((s, \langle) \rightarrow (t, \langle, R)), \quad ((t, \rangle) \rightarrow (\text{accept}, \rangle, L)), \\ &((t, x) \rightarrow (t, x, R)), \quad ((t, a) \rightarrow (u, x, R)), \\ &((u, a) \rightarrow (u, a, R)), \quad ((u, x) \rightarrow (u, x, R)), \\ &((u, b) \rightarrow (v, x, R)), \quad ((v, b) \rightarrow (v, b, R)), \\ &((v, x) \rightarrow (v, x, R)), \quad ((v, c) \rightarrow (w, x, L)), \\ & \quad \quad \quad ((w, b) \rightarrow (w, b, L)), \\ &((w, a) \rightarrow (w, a, L)), \quad ((w, x) \rightarrow (w, x, L)), \\ &((w, \langle) \rightarrow (t, \langle, R))\} \end{aligned}$$

# ALL no Jflap

$MT = (\{s, t, u, v, w, \text{accept}\}, \{a, b, c\}, \{a, b, c, X\}, s, \delta, \{\text{accept}\}, \langle, \rangle)$



$\delta$

Não é tão eficiente como a primeira máquina

Input	Result
<abc>	Accept
<aabbcc>	Accept
<abbcc>	Reject
<>	Accept
<aabc>	Reject

Em cada passo casamos um a, b, c e trocamos cada um deles por X até não haver mais triplas.

Cada estado é explicado a seguir:

- State t looks for the leftmost a, changes this to an x, and moves into state u. If no symbol from the input alphabet can be found, then the input string is accepted.
- State u moves right past any a's or x's until it finds a b. It changes this b to an x, and moves into state v.
- State v moves right past any b's or x's until it finds a c. It changes this c to an x, and moves into state w.
- State w moves left past any a's, b's or x's until it reaches the start boundary, and moves into state t.

# CSG = LBA

- Uma linguagem é aceita por um LBA sse é gerada por uma GSC.
- Lembrem que as GLC não são tão poderosas para descrever todos os aspectos de uma linguagem de programação,
  - entretanto, um ALL é muito caro: tempo exponencial... portanto não são usados no contexto de compiladores.

# Definição de Menezes para LBA

MT com fita limitada é uma óctupla  
 $(Q, \Sigma, \Gamma, \delta, q_0, F, \langle, \rangle)$

Em que  $\delta : Q \times (\Gamma \cup \{\langle, \rangle\}) \rightarrow$   
 $2^{Q \times (\Gamma \cup \{\langle, \rangle\}) \times \{L, R\}}$

$\delta(p, x) = \{(q_1, y_1, m_1), \dots, (q_n, y_n, m_n)\}$

# Exercícios

1. Faça uma MT que reconheça  $L = \{0^n 1^n \mid n \geq 1\} \rightarrow$  LLC
2. Faça uma MT que reconheça  $L = \{w\#w \mid w \in \{0,1\}^*\} \rightarrow$  LSC
3. Faça uma MT que reconheça  $L = \{0^{2^n} \mid n \geq 0\}$  cadeias de 0's cujo tamanho é potência de 2  $\rightarrow$  foi gerada por uma Gramática Irrestrita, no início do curso
4. Faça uma MT que reconheça  $L = \{x \mid x \in \{a,b,c\}^* \text{ e } x \text{ é uma permutação de } a^n b^n c^n \text{ para algum } n \geq 0\}$  LSC
5. Considere  $L = \{0^n 1^m 0^n \mid n, m > 0\}$ . Qual é o tipo da linguagem? Justifique. Escreva um ALL para processar L.