

# Linguagens Formais e Autômatos

Hierarquia de Chomsky:

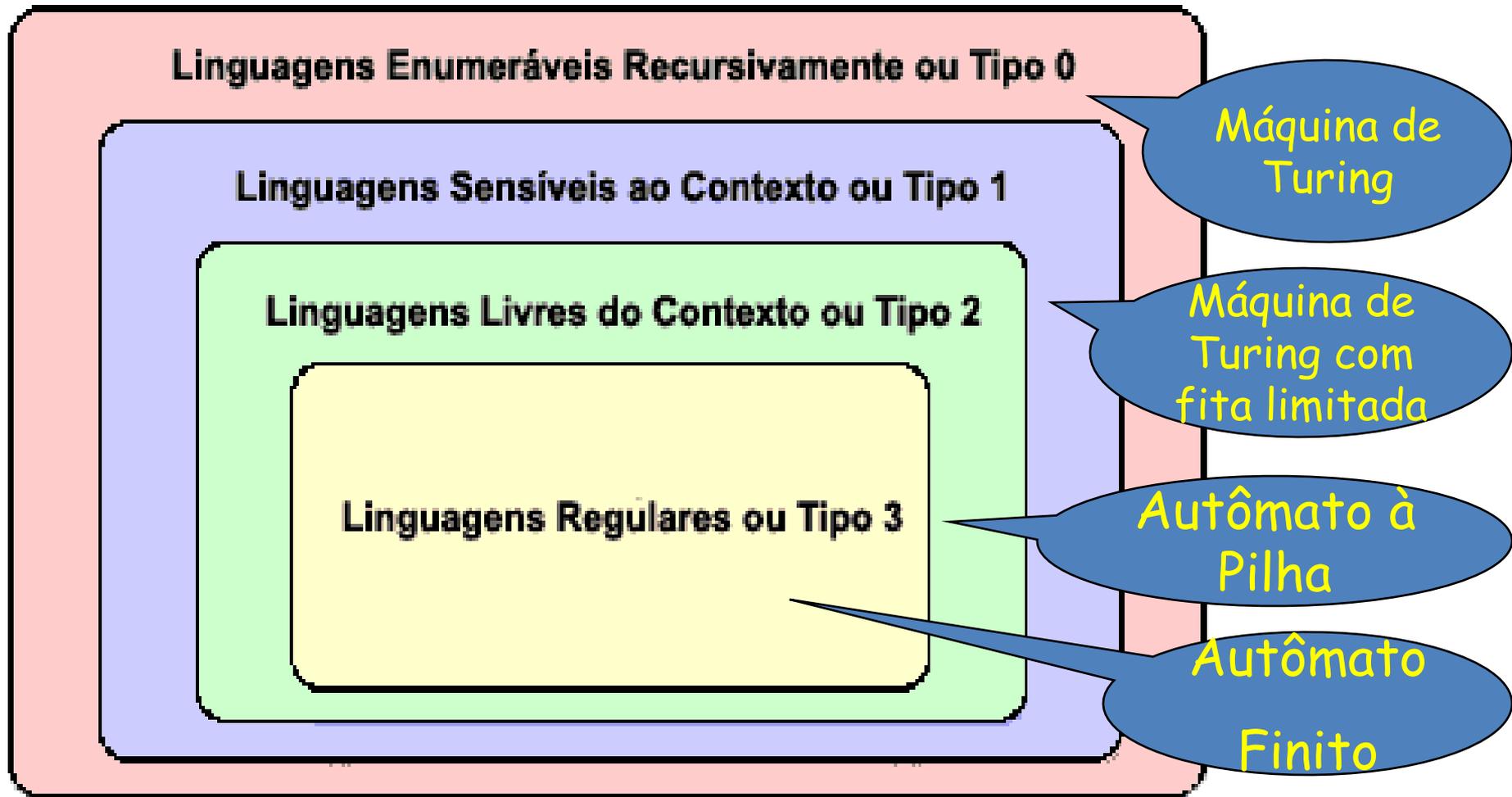
Linguagens Regulares, Livres de Contexto, Sensíveis ao Contexto e Irrestritas  
seus Reconhecedores, e Geradores

Autômatos são essenciais para o estudo dos limites da computação. Existem  
dois limites estudados aqui:

O que um computador pode fazer?

O que um computador pode fazer eficientemente?

# Hierarquia de Chomsky e sua relação com os autômatos



# LFA e a Teoria da Computabilidade

Todas as linguagens recursivas são RE.

Todas as regulares, livres de contexto e sensíveis ao contexto são recursivas

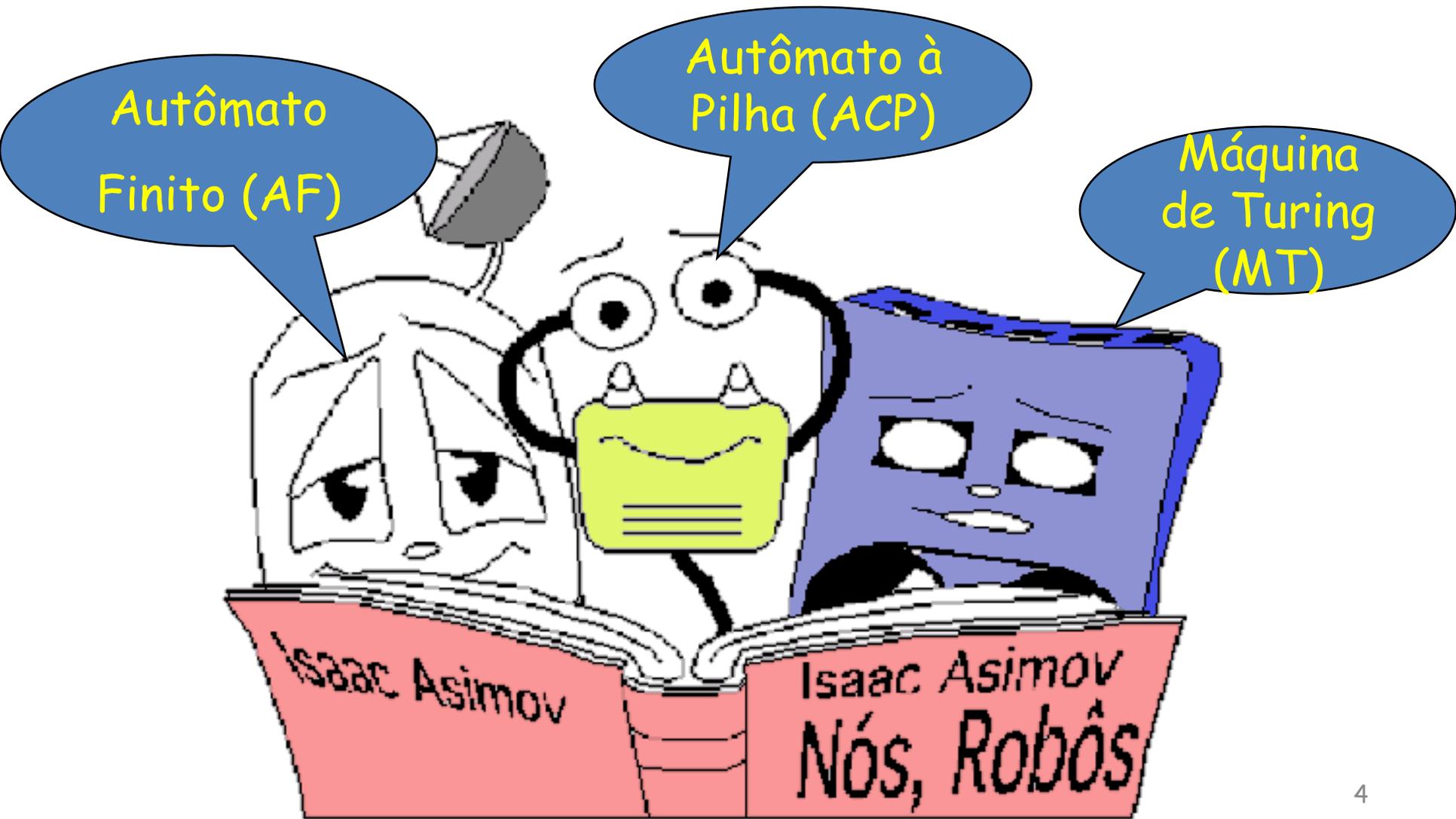
Máquina de Turing que reconhece

Automata theory: formal languages and formal grammars			
Chomsky hierarchy	Grammars	Languages	Minimal automaton
Type-0	(unrestricted)	Recursively enumerable	Turing machine
	(unrestricted)	<b>Recursive</b>	Decider
Type-1	Context-sensitive	Context-sensitive	Linear-bounded
Type-2	Context-free	Context-free	Pushdown
Type-3	Regular	Regular	Finite

Each category of languages or grammars is a proper superset of the category directly beneath it.

Máquina de Turing que Decide

# Autômatos Estudados



# Exemplos de AF's

- Pensem em:
- um controlador automático de portas de shoppings
- um controlador de elevador
- um interruptor
  
- AF tem um número FINITO de estados, cujo propósito é lembrar de porções relevantes da história do sistema.
  - A história inteira não pode ser lembrada ...uma decisão importante de projeto é o que os estados significam.

# JFLAP

<http://www.jflap.org>

<http://pt.wikihow.com/Baixar-e-Usar-o-JFLAP>

JFLAP  
Version 7.0  
RELEASED  
August 28,  
2009

Não  
veremos



Autômatos

Gramáticas

Expressões Regulares -

denotam linguagens  
regulares

# Usando JFLAP com AF: interruptor “onofre”

The screenshot shows the JFLAP software interface. At the top, there is a menu bar with 'File', 'Input', 'Test', 'Convert', and 'Help'. Below the menu bar is the 'Editor' tab, which contains a finite automaton diagram. The diagram has two states: 'q0' (labeled 'off') and 'q1' (labeled 'on'). There is a transition from 'q0' to 'q1' labeled 'ligar' and a transition from 'q1' to 'q0' labeled 'desligar'. Below the diagram is a window titled 'Accepting configuration found!' which displays a list of configurations: 'q0', 'q1', 'q0', and 'q1', each with the input string 'ligar""desligar""ligar' and a downward arrow. At the bottom of this window are two buttons: 'Keep looking' and 'I'm done'. A blue speech bubble points to the input string in the 'Accepting configuration found!' window, containing the text 'Entrada: ligar""desligar""ligar'. Another blue speech bubble points to the bottom of the JFLAP window, containing the text 'representam as informações externas sobre o sistema'.

Como a história inteira não pode ser lembrada temos que escolher o que é importante esquecer e o que não é!

JFLAP : <untitled2>

File Input Test Convert Help

Editor Simulate: ligardesligarligar

```
graph LR; start(( )) --> q0((q0)); q0 -- ligar --> q1(((q1))); q1 -- desligar --> q0;
```

q0

ligar

desligar

q1

q1

ligardesligarligar

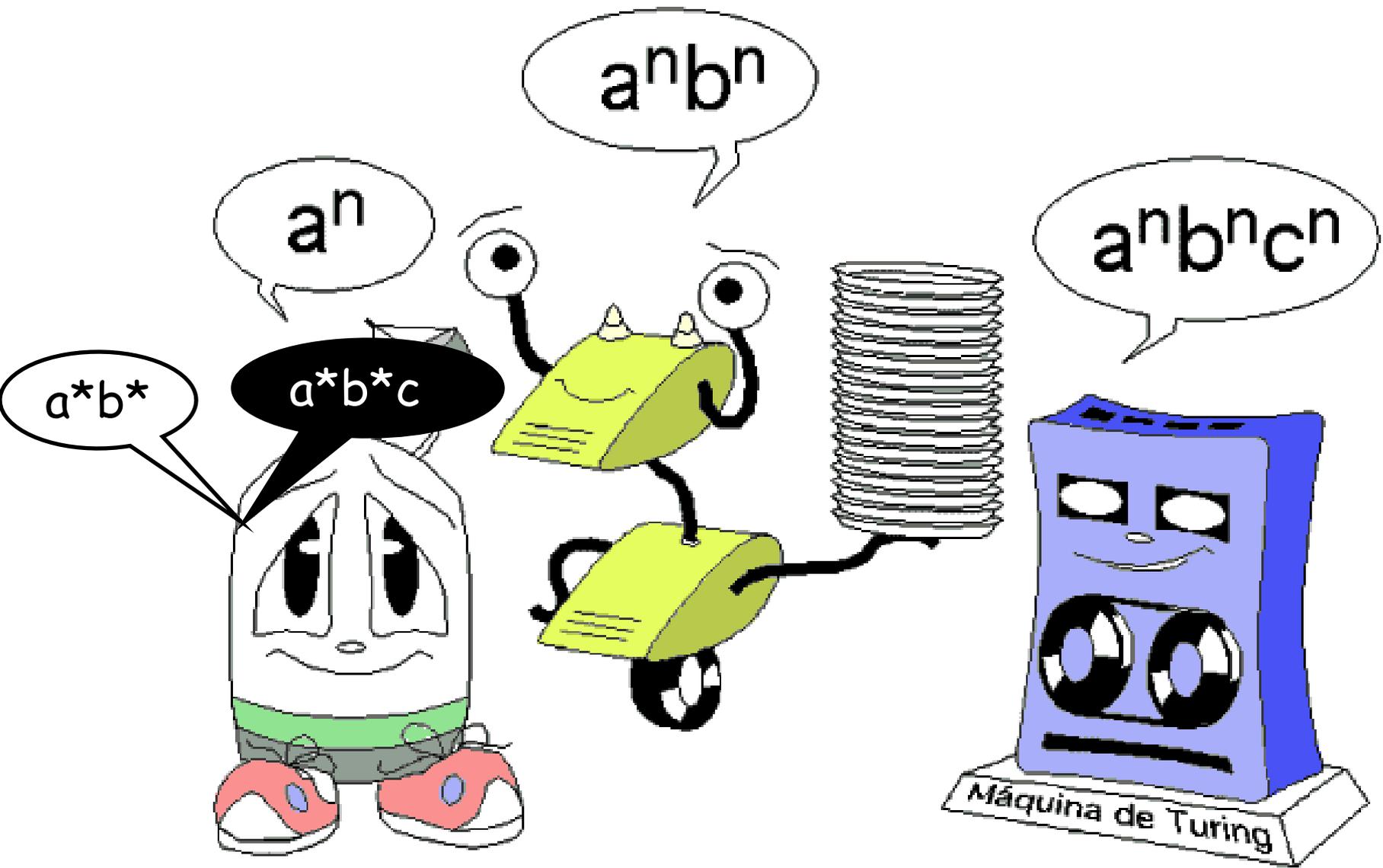
Step Reset Freeze Thaw Trace Remove

Java Applet Window

Iniciat aula1\_jfa.ppt JFLAP - Microsoft Intern... JFLAP : <untitled2>

11:42

Transições sem aspas - entrada fica confusa



# Linguagens Abstratas?

## Relação das Linguagens Formais com a Definição de LPs

1)  $L1 = \{a^n b^n \mid n \geq 0\}$

2)  $L2 = \{wcw \mid w \in \{a,b\}^*\}$  ex: aabcaab

3)  $L3 = \{a^n b^m c^n d^m \mid n \geq 1 \text{ e } m \geq 1\}$

$$1) L1 = \{a^n b^n \mid n \geq 0\}$$

Esta linguagem é livre de contexto e é análoga às linguagens que permitem duplo balanceamento:

- a) Linguagens bloco-estruturadas do tipo **BEGIN<sup>n</sup>**  
**END<sup>n</sup>**
- b) Linguagens com parênteses balanceados na forma  
**(<sup>n</sup>)<sup>n</sup>**

2)  $L2 = \{wcw \mid w \in \{a,b\}^*\}$   
ex: aabcaab

Esta linguagem não é livre de contexto e ela abstrai o problema de **checar se os identificadores foram declarados antes de serem usados no corpo de um programa Pascal**, por exemplo.

$$3) L3 = \{a^n b^m c^n d^m \mid n \geq 1 \text{ e } m \geq 1\}$$

Esta linguagem não é livre de contexto e ela abstrai o problema de **checar se o número de parâmetros formais na declaração de um procedimento concorda com o número de parâmetros reais no uso do procedimento.**

$a^n b^m$  poderiam representar a lista de par. formais em 2 proc. declarados com  $n$  e  $m$  argumentos e  $c^n d^m$  a lista de par reais na chamada destes 2 procedimentos.

# Aplicações Práticas

- **AF/Linguagens Regulares/Expressões Regulares:**
  - Analisador léxico (Scanner) de um **compilador**
  - Casamento de padrões em utilitários de busca
  - Verificação de protocolos de comunicação ou protocolos para troca segura de informação na Internet
  - Projeto e verificação de circuitos digitais
- **APilha/Linguagens Livre de Contexto**
  - Analisador sintático (Parser) de um **compilador**
  - Linguagens de descrição de documentos (DTD's ou schemas) como SGML, XML.

# Estrutura de um Compilador

-----Front-End -----

