

00277957

# Gramáticas Irrestritas

Isis Burmeister Pericolo

# Histórico

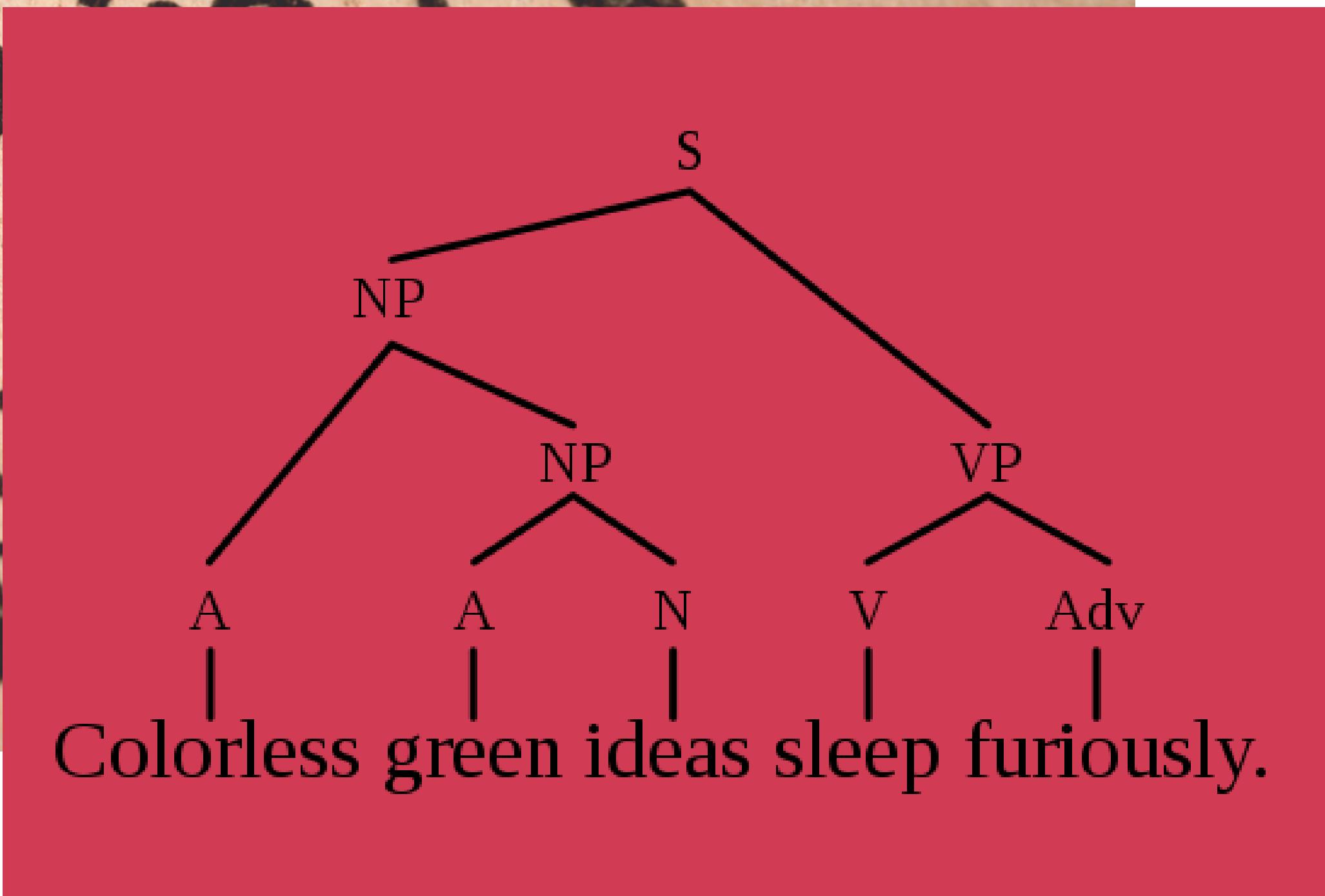
O modelo das gramáticas irrestritas foi inicialmente discutido por **Noam Chomsky** em dois artigos - um em **1956** (Three models for the description of language) e o outro em **1959** (On certain formal properties of grammars) - que propunham a Hierarquia de Chomsky.

# Histórico

O modelo das gramáticas irrestritas foi inicialmente discutido por Noam Chomsky em dois artigos - um em 1956 (*Three models for the description of language*) e o outro em 1959 (*On certain formal properties of grammars*) - que propunham a Hierarquia de Chomsky.

Estes **definiam classes de gramáticas formais**, ainda que tivessem ênfase no tratamento e classificação de linguagens naturais mais do que linguagens formais.

lang! tēr-ə-190'', 'ŋ  
lañ! guñ  
gage;  
tong  
1.



# Formalização

Uma gramática irrestrita é uma gramática  $G = (N, \Sigma, P, S)$  na qual

# Formalização

Uma gramática irrestrita é uma gramática  $G = (N, \Sigma, P, S)$  na qual

- N é um conjunto de símbolos não-terminais

# Formalização

Uma gramática irrestrita é uma gramática  $G = (N, \Sigma, P, S)$  na qual

- $N$  é um conjunto de símbolos não-terminais
- $\Sigma$  é um conjunto de símbolos terminais

# Formalização

Uma gramática irrestrita é uma gramática  $G = (N, \Sigma, P, S)$  na qual

- $N$  é um conjunto de símbolos não-terminais
- $\Sigma$  é um conjunto de símbolos terminais
- $P$  é um conjunto de regras de produção  $\alpha \rightarrow \beta$ , onde
  - $\alpha$  e  $\beta$  são strings de símbolos em  $N \cup \Sigma$
  - $\alpha$  não é uma string vazia

# Formalização

Uma gramática irrestrita é uma gramática  $G = (N, \Sigma, P, S)$  na qual

- $N$  é um conjunto de símbolos não-terminais
- $\Sigma$  é um conjunto de símbolos terminais
- $P$  é um conjunto de regras de produção  $\alpha \rightarrow \beta$ , onde
  - $\alpha$  e  $\beta$  são strings de símbolos em  $N \cup \Sigma$
  - $\alpha$  não é uma string vazia
- $S \in N$  é um símbolo de início

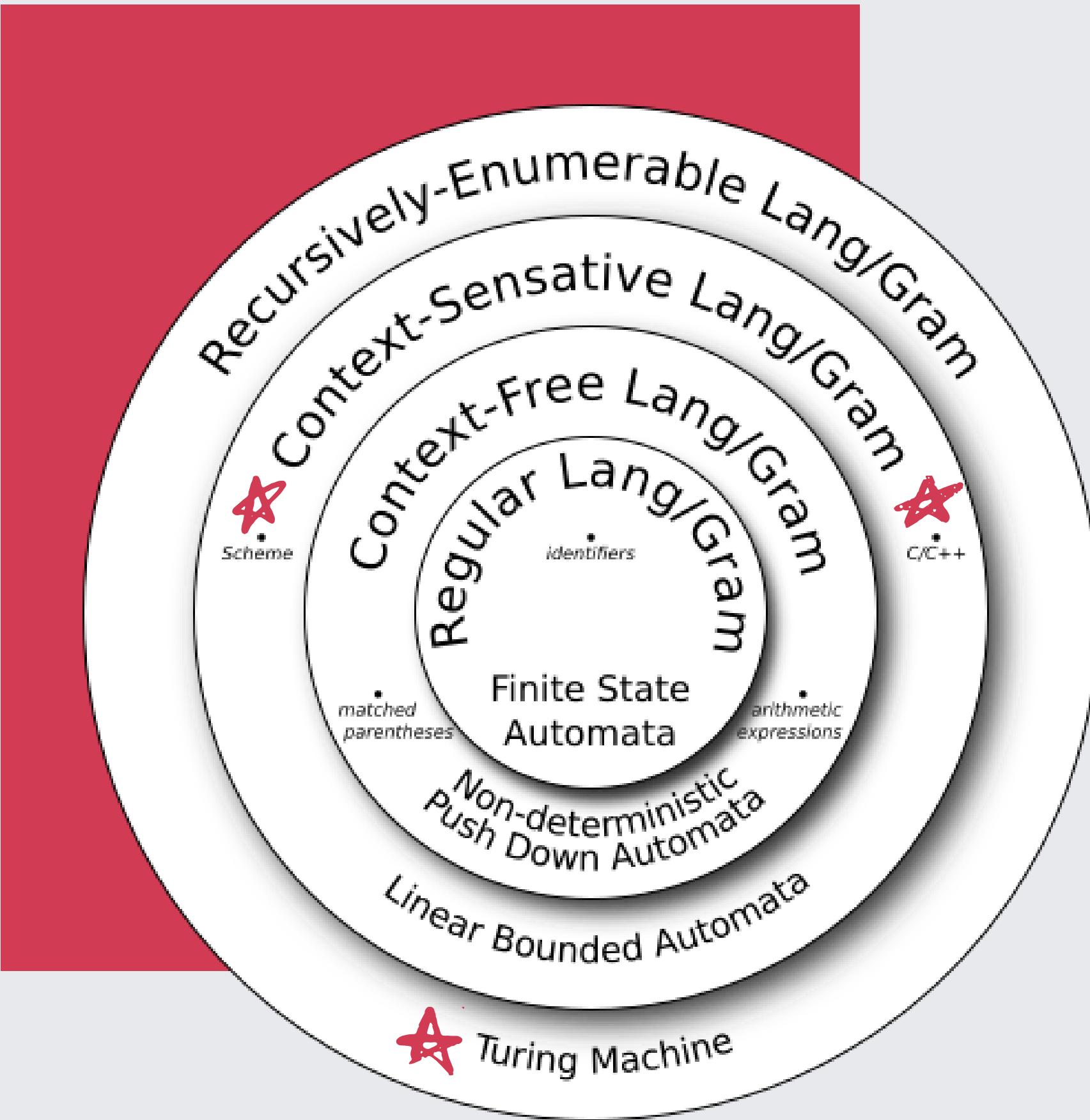
# Formalização

É comum na área de ciência da computação falarmos de **linguagens recursivamente enumeráveis** - que é a classe de linguagens formais que podem ser geradas a partir de gramáticas irrestritas.

# Formalização

É comum na área de ciência da computação falarmos de **linguagens recursivamente enumeráveis** - que é a classe de linguagens formais que podem ser geradas a partir de gramáticas irrestritas.

Alguns autores, incluindo o próprio Chomsky, ainda podem usar outros nomes para identificar GIs, como: **gramáticas tipo-0**, **gramáticas de estrutura frasal**, e até mesmo **semi-Thue**.



## HIERARQUIA DE CHOMSKY



# Exemplos

$$L = \{a^n b^n c^n \mid n \geq 1\}$$

1

# Exemplos

1

$$L = \{a^n b^n c^n \mid n \geq 1\}$$

Uma gramática irrestrita que gera a linguagem descrita pode ser:

$$P \rightarrow aPBc \mid abc$$

$$cB \rightarrow Bc$$

$$bB \rightarrow bb$$

# Exemplos

$$L = \{a^n b^n c^n \mid n \geq 1\}$$

Uma possível derivação para esta linguagem, considerando  $n = 3$  é:

1

$$\begin{array}{ll} P \Rightarrow aP Bc & \Rightarrow aaabBcBcc \\ \hline & \Rightarrow aaab\underline{B}Bccc \\ \Rightarrow aa\underline{a}P BcBc & \Rightarrow aaab\underline{bb}Bccc \\ \hline & \Rightarrow aaabbb\underline{c}cc \\ \Rightarrow aaabcBcBc & \Rightarrow aaabbbccc \end{array}$$

# Exemplos

$L = \{ww \mid w \in (a, b)^*\}$

2

# Exemplos

2

$$L = \{ww \mid w \in (a, b)^*\}$$

Uma gramática irrestrita que gera a linguagem descrita pode ser:

$$S \rightarrow X\#$$

$$A\# \rightarrow A'\#$$

$$B\# \rightarrow B'\#$$

$$X \rightarrow aXA \mid bXb \mid \#$$

$$AA' \rightarrow A'A$$

$$AB' \rightarrow B'A$$

$$\#\# \rightarrow \epsilon$$

$$BA' \rightarrow A'B$$

$$BB' \rightarrow B'B$$

$$\#A' \rightarrow a\#$$

$$\#B' \rightarrow b\#$$

# Exemplos

$$L = \{ww \mid w \in (a, b)^*\}$$

Considerando  $w = abb$ , podemos derivar esta linguagem:

2

$$\begin{array}{lll} \underline{P} \Rightarrow \underline{X\#} & \Rightarrow abb\#\underline{BA'B\#} & \Rightarrow abbab\#\underline{B'\#} \\ \Rightarrow a\underline{XA\#} & \Rightarrow abb\#\underline{A'BB\#} & \Rightarrow abbab\underline{bb\#\#} \\ \Rightarrow ab\underline{XA}B\# & \Rightarrow abba\#\underline{BB\#} & \Rightarrow abbab\underline{b}\ddagger \\ \Rightarrow abb\underline{X}BBA\# & \Rightarrow abba\#\underline{BB'}\# & \\ \Rightarrow abb\#\underline{B}BA\# & \Rightarrow abba\#\underline{B'}B\# & \\ \Rightarrow abb\#\underline{B}BA'\# & \Rightarrow abbab\#\underline{B\#} & \end{array}$$

# Turing-Completude

Gramáticas irrestritas, **por definição**, sempre produzem linguagens que podem ser reconhecidas e descritas por máquinas de Turing. Embora um modelo seja gerador e o outro seja reconhecedor, é possível transformar GIs em MTs e vice-versa.

# Turing-Completude

Gramáticas irrestritas, por definição, sempre produzem linguagens que podem ser reconhecidas e descritas por máquinas de Turing. Embora um modelo seja gerador e o outro seja reconhecedor, é possível transformar GIs em MTs e vice-versa.

Não apenas isso, mas as **GIs e MTs equivalentes terão computações idênticas**, visto que seguem um mesmo conjunto de produções, apenas fazendo-as na ordem inversa uma da outra.

# Turing-Completude

## DE GRAMÁTICAS IRRESTRITAS PARA MÁQUINAS DE TURING

Para passarmos de uma gramática para uma máquina de Turing, geralmente usamos uma **máquina não determinística de duas fitas**, na qual em uma geraremos as cadeias de linguagens e usaremos a saída dela como entrada para a segunda fita, que comparará esta com a palavra **w**.

# Turing - Completude

## DE GRAMÁTICAS IRRESTRITAS PARA MÁQUINAS DE TURING

Para passarmos de uma gramática para uma máquina de Turing, geralmente usamos uma máquina não determinística de duas fitas, na qual em uma geraremos as cadeias de linguagens e usaremos a saída dela como entrada para a segunda fita, que comparará esta com a palavra **w**.

A partir disso podemos aceitar todas as palavras geradas pela gramática irrestrita original, embora isso possa ser um processo taxativo e demorado dependendo de sua definição.



o o o



o o o

## GERA PALAVRAS DA LINGUAGEM

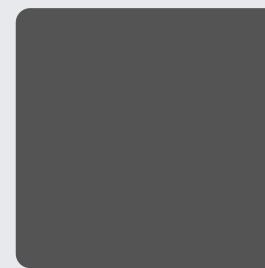
Regras de transição geram todas as possíveis palavras da linguagem não-deterministicamente.



W



ooo



ooo

## GERA PALAVRAS DA LINGUAGEM

Regras de transição geram todas as possíveis palavras da linguagem não-deterministicamente.



## COMPARA COM PALAVRA W

Programada para comparar a entrada vinda da fita de cima com a palavra w.



w



ooo



ooo



ooo

## GERA PALAVRAS DA LINGUAGEM

Regras de transição geram todas as possíveis palavras da linguagem não-deterministicamente.



## COMPARA COM PALAVRA W

Programada para comparar a entrada vinda da fita de cima com a palavra w.



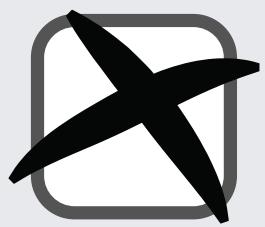
w



ooo



ou



# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

Partiremos de uma descrição de máquina de Turing tal que

$x_j \dots x_k q_n x_l \dots x_m$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

Partiremos de uma descrição de máquina de Turing tal que

$$x_j \dots x_k q_n x_l \dots x_m$$

E sabemos que uma MT aceita uma palavra de uma linguagem se

$$q_0 w \Rightarrow^* w_1 q_a w_2$$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

Partiremos de uma descrição de máquina de Turing tal que

$$x_j \dots x_k q_n x_l \dots x_m$$

E sabemos que uma MT aceita uma palavra de uma linguagem se

$$q_0 w \Rightarrow^* w_1 q_a w_2$$

Enquanto uma GI gera uma palavra de uma linguagem se

$$S \Rightarrow^* w$$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Iniciação.** Estas produções constroem uma cadeia onde **B** indica um branco e **&** é uma variável especial usada para terminação;  $B\&w_1q_aw_2B$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Iniciação.** Estas produções constroem uma cadeia onde **B** indica um branco e **&** é uma variável especial usada para terminação;  $B\&w_1q_aw_2B$
- **Execução.** Para cada regra de transição  **$\delta$**  necessita-se uma produção correspondente;

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Iniciação.** Estas produções constroem uma cadeia onde **B** indica um branco e **&** é uma variável especial usada para terminação;  $B\&w_1q_aw_2B$
- **Execução.** Para cada regra de transição  **$\delta$**  necessita-se uma produção correspondente;
- **Limpeza.** A derivação deixará alguns símbolos **q0**, **B** e **&** na cadeira, então são necessárias algumas produções adicionais para limpá-los e deixar apenas a cadeia final.

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Iniciação.** Estas produções constroem uma cadeia onde **B** indica um branco e **&** é uma variável especial usada para terminação;

---

$$B \dots B \& w_1 q_a w_2 B \dots B$$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Iniciação.** Estas produções constroem uma cadeia onde **B** indica um branco e **&** é uma variável especial usada para terminação;

$$\frac{B \dots B \& w_1 q_a w_2 B \dots B}{}$$

$$S \rightarrow BS \mid SB \mid \& A$$

# Turing - Completude

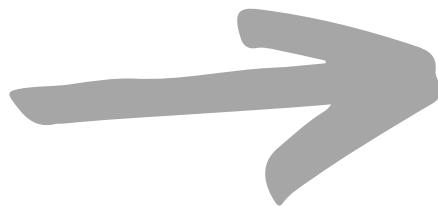
## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Iniciação.** Estas produções constroem uma cadeia onde **B** indica um branco e **&** é uma variável especial usada para terminação;

$$\frac{B \dots B \& w_1 q_a w_2 B \dots B}{}$$



$$S \rightarrow BS \mid SB \mid \& A$$



Para gerar um número arbitrário de símbolos vazios **B** e iniciar a sequência.

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Iniciação.** Estas produções constroem uma cadeia onde **B** indica um branco e **&** é uma variável especial usada para terminação;

$$\frac{B \dots B \& w_1 q_a w_2 B \dots B}{}$$



$$A \rightarrow xA \mid Ax \mid q_a, \forall x \in \Sigma$$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Iniciação.** Estas produções constroem uma cadeia onde **B** indica um branco e **&** é uma variável especial usada para terminação;

$$\frac{B \dots B \& w_1 q_a w_2 B \dots B}{}$$

$$A \rightarrow xA \mid Ax \mid q_a, \forall x \in \Sigma$$


Para gerar as cadeias **w1** e **w2** com um estado **qa** em algum lugar no meio.

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Execução.** Para cada regra de transição  $\delta$  necessita-se uma produção correspondente;

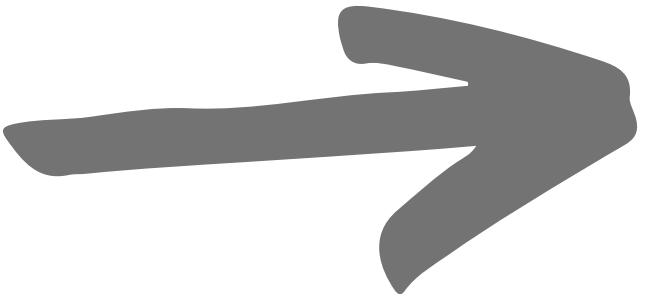
$$\delta(q_m, a) = (q_n, b, D)$$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Execução.** Para cada regra de transição  $\delta$  necessita-se uma produção correspondente:

$$\delta(q_m, a) = (q_n, b, D)$$



$$b q_n \rightarrow q_m a$$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Execução.** Para cada regra de transição  $\delta$  necessita-se uma produção correspondente;

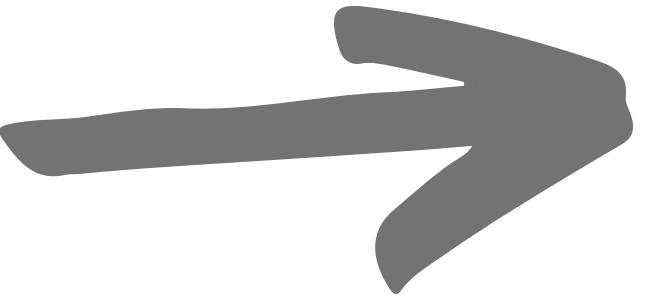
$$\delta(q_m, a) = (q_n, b, E)$$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Execução.** Para cada regra de transição  $\delta$  necessita-se uma produção correspondente:

$$\delta(q_m, a) = (q_n, b, E)$$



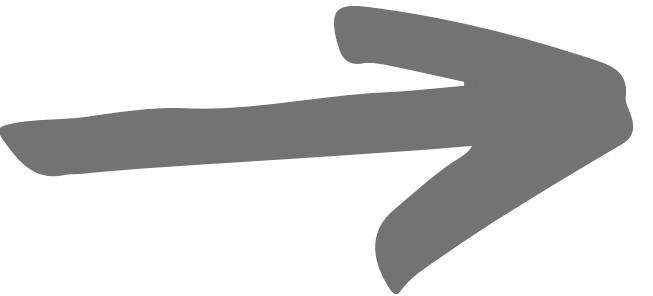
$$q_n c b \rightarrow c q_m a$$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Execução.** Para cada regra de transição  $\delta$  necessita-se uma produção correspondente:

$$\delta(q_m, a) = (q_n, b, E)$$



$$q_n c b \rightarrow c q_m a$$

A assimetria é devido ao fato do símbolo à direita de  $q$  ser o símbolo sob a cabeça de leitura/escrita da máquina de Turing.

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Limpeza.** A derivação deixará alguns símbolos  $q_0$ ,  $B$  e  $\&$  na cadeira, então são necessárias algumas produções adicionais para limpá-los e deixar apenas a cadeia final.

$B \dots B \& q_0 w B \dots B$

# Turing - Completude

## DE MÁQUINAS DE TURING PARA GRAMÁTICAS IRRESTRITAS

- **Limpeza.** A derivação deixará alguns símbolos  $q_0$ ,  $B$  e  $\&$  na cadeira, então são necessárias algumas produções adicionais para limpá-los e deixar apenas a cadeia final.

$B \dots B\&q_0wB\dots B$



$B \rightarrow \varepsilon$

$\&q_0 \rightarrow \varepsilon$

”

For one thing, studying language is by itself a part of a study of human intelligence that is, perhaps, the central aspect of human nature.

- Noam Chomsky