ACH2043
INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

Aula 12

Cap 3.2 – Variantes de Máq. de Turing
Extra: Linguagens Sensíveis ao Contexto

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

Máquinas de Turing – Definição formal

Uma ***máquina de Turing*** é uma 7-upla, $(Q, \Sigma, \Gamma, \delta, q_0, q_{aceita}, q_{rejeita})$, onde Q, Σ, Γ são todos conjuntos finitos e

1. Q é o conjunto de estados,
2. Σ é o alfabeto de entrada sem o ***símbolo em branco*** $\sqcup$,
3. Γ é o alfabeto de fita, onde $\sqcup \in \Gamma$ e $\Sigma \subseteq \Gamma$,
4. $\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{E, D\}$ é a função de transição,
5. $q_0 \in Q$ é o estado inicial,
6. $q_{aceita} \in Q$ é o estado de aceitação, e
7. $q_{rejeita} \in Q$ é o estado de rejeição, onde $q_{rejeita} \neq q_{aceita}$.

Máquinas de Turing Não-Determinísticas

Máquinas de Turing Não-Determinísticas

$$\delta : Q \times \Gamma \longrightarrow \mathcal{P}(Q \times \Gamma \times \{\mathrm{E}, \mathrm{D}\}).$$

TEOREMA 3.16

Toda máquina de Turing não-determinística tem uma máquina de Turing determinística que lhe é equivalente.

TEOREMA 3.16

Toda máquina de Turing não-determinística tem uma máquina de Turing determinística que lhe é equivalente.

Ideia da prova:

Simular uma MTND N por uma MTD D

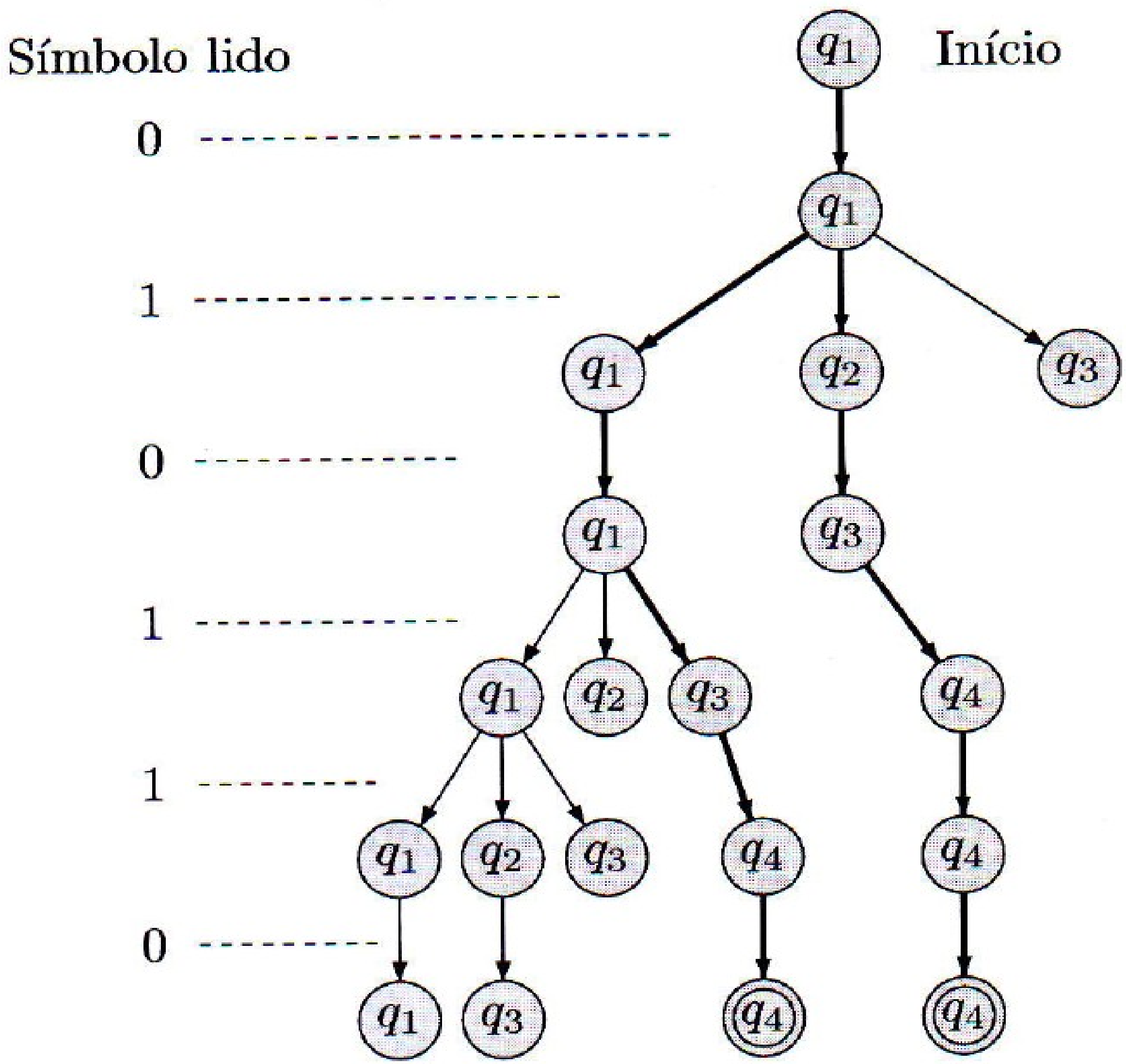
Computação de N representada por uma árvore (filhos de um nó são as possibilidades de transição)

Cada caminho (a partir da raiz) é uma computação possível)

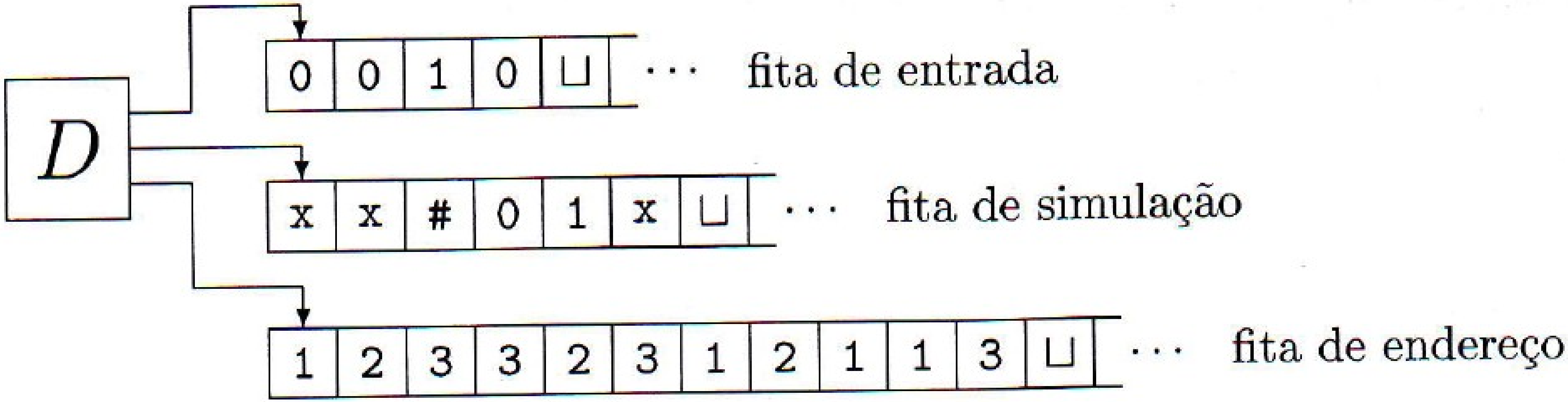
Cada nó dessa árvore terá um endereço que indica qual filho seguir a partir da raiz (ex: 314)

D irá percorrer essa árvore através de busca em largura (para impedir que caia em um ramo infinito)

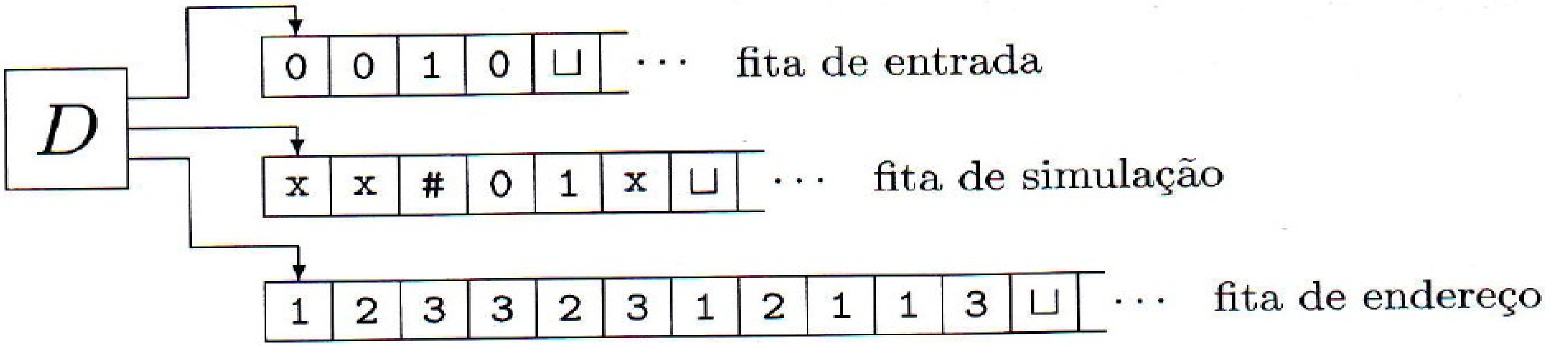
Lembram da árvore para AFNDs?



Prova


D
0 0 1 0 ⊔ ···
fita de entrada
x x # 0 1 x ⊔ ···
fita de simulação
1 2 3 3 2 3 1 2 1 1 3 ⊔ ···
fita de endereço

Prova

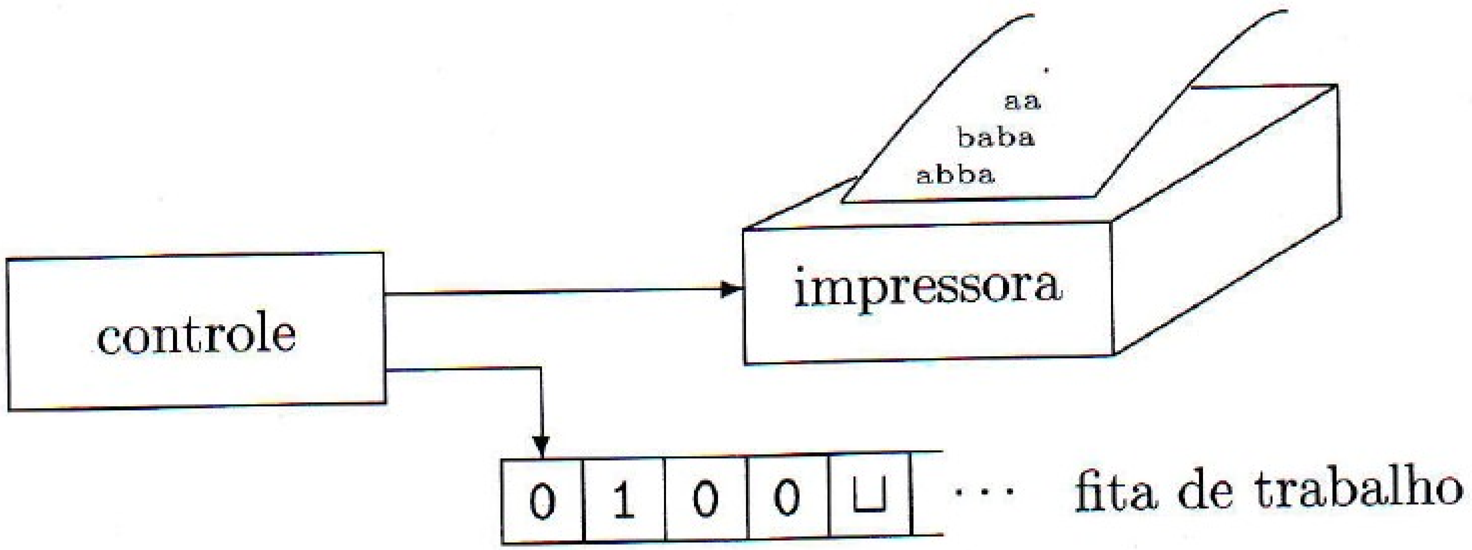


1. Inicialmente, a fita 1 contém a entrada w e as fitas 2 e 3 estão vazias.
2. Copie a fita 1 para a fita 2.
3. Use a fita 2 para simular N com a entrada w sobre um ramo de sua computação não-determinística. Antes de cada passo de N, consulte o próximo símbolo na fita 3 para determinar qual escolha fazer entre aquelas permitidas pela função de transição de N. Se não restam mais símbolos na fita 3 ou se essa escolha não-determinística for inválida, aborte esse ramo indo para o estágio 4. Também vá para o estágio 4 se uma configuração de rejeição for encontrada. Se uma configuração de aceitação for encontrada, *aceite* a entrada.
4. Substitua a cadeia na fita 3 pela próxima cadeia na ordem lexicográfica. Simule o próximo ramo da computação de N indo para o estágio 2.

COROLÁRIO 3.18

Uma linguagem é Turing-reconhecível se e somente se alguma máquina de Turing não-determinística a reconhece.

Enumeradores



Duas fitas, sendo uma só de escrita (impressora)

Começa com a fita impressora em branco

Imprime cadeias da linguagem, em qualquer ordem, possivelmente com repetição

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (<=)

Temos um enumerador E

M funciona assim: dada uma cadeia w

•

•

•

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (<=)

Temos um enumerador E

M funciona assim: dada uma cadeia w

- roda E e compara a cadeia enumerada com w
- se for igual, aceita w
- rejeita w se acabar a enumeração

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (<=)

Temos um enumerador E

M funciona assim: dada uma cadeia w

- roda E e compara a cadeia enumerada com w
- se for igual, aceita w
- rejeita w se acabar a enumeração

Se w pertence a L, será aceita

Se w não pertence a L,

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (<=)

Temos um enumerador E

M funciona assim: dada uma cadeia w

- roda E e compara a cadeia enumerada com w
- se for igual, aceita w
- rejeita w se acabar a enumeração

Se w pertence a L, será aceita

Se w não pertence a L,

M rejeita se L for finita

M não pára se L for infinita

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (=>) Temos uma Máquina de Turing M

O enumerador E funciona assim:

TEOREMA 3.21 ....

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (=>) Temos uma Máquina de Turing M

O enumerador E funciona assim:

Seja s_1, s_2, s_3, … sequências de Σ^* (ordem crescente de comprimento e ordem lexicográfica)

-
-

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (=>) Temos uma Máquina de Turing M

O enumerador E funciona assim:

Seja s_1, s_2, s_3, … sequências de Σ^* (ordem crescente de comprimento e ordem lexicográfica)

- Ignore a entrada
- para i = 1, 2, 3, …
  - Rode M sobre s_i
  - Imprima s_i se for aceita

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (=>) Temos uma Máquina de Turing M

O enumerador E funciona assim:

Seja s_1, s_2, s_3, … sequências de Σ^* (ordem crescente de comprimento e ordem lexicográfica)

- Ignore a entrada
- para i = 1, 2, 3, …

  Rode M sobre s_i

  Imprima s_i se for aceita

Problema?

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (=>) Temos uma Máquina de Turing M

O enumerador E funciona assim:

Seja s_1, s_2, s_3, … sequências de Σ^* (ordem crescente de comprimento e ordem lexicográfica)

- Ignore a entrada
- para i = 1, 2, 3, …

  Rode M sobre s_i

  Imprima s_i se for aceita

Problema?

E se M entrar em loop?

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (=>) Temos uma Máquina de Turing M

O enumerador E funciona assim:

Seja s_1, s_2, s_3, … sequências de Σ^* (ordem crescente de comprimento e ordem lexicográfica)

- Ignore a entrada
- para i = 1, 2, 3, …

  Rode M sobre s_i

  Imprima s_i se for aceita

Problema?

E se M entrar em loop? Travará o enumerador...

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (=>) Temos uma Máquina de Turing M

O enumerador E funciona assim:

Seja s_1, s_2, s_3, … sequências de Σ^* (ordem crescente de comprimento e ordem lexicográfica)

- ~~Ignore a entrada~~
- ~~para i = 1, 2, 3, …~~

  ~~Rode M sobre s_i~~

  ~~Imprima s_i se for aceita~~

Problema?

E se M entrar em loop? Travará o enumerador...

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (=>) Temos uma Máquina de Turing M

O enumerador E funciona assim:

Seja s_1, s_2, s_3, … sequências de Σ^* (ordem crescente de comprimento e ordem lexicográfica)

- Ignore a entrada
- para i = 1, 2, 3, …

  Rode i passos de M sobre $s_1, ..., s_i$

  Imprima as s_j que foram aceitas

TEOREMA 3.21

Uma linguagem é Turing-reconhecível se e somente se algum enumerador a enumera.

Prova: (=>) Temos uma Máquina de Turing M

O enumerador E funciona assim:

Seja s_1, s_2, s_3, … sequências de Σ^* (ordem crescente de comprimento e ordem lexicográfica)

- Ignore a entrada
- para i = 1, 2, 3, …

  Rode i passos de M sobre s_1, ..., s_i

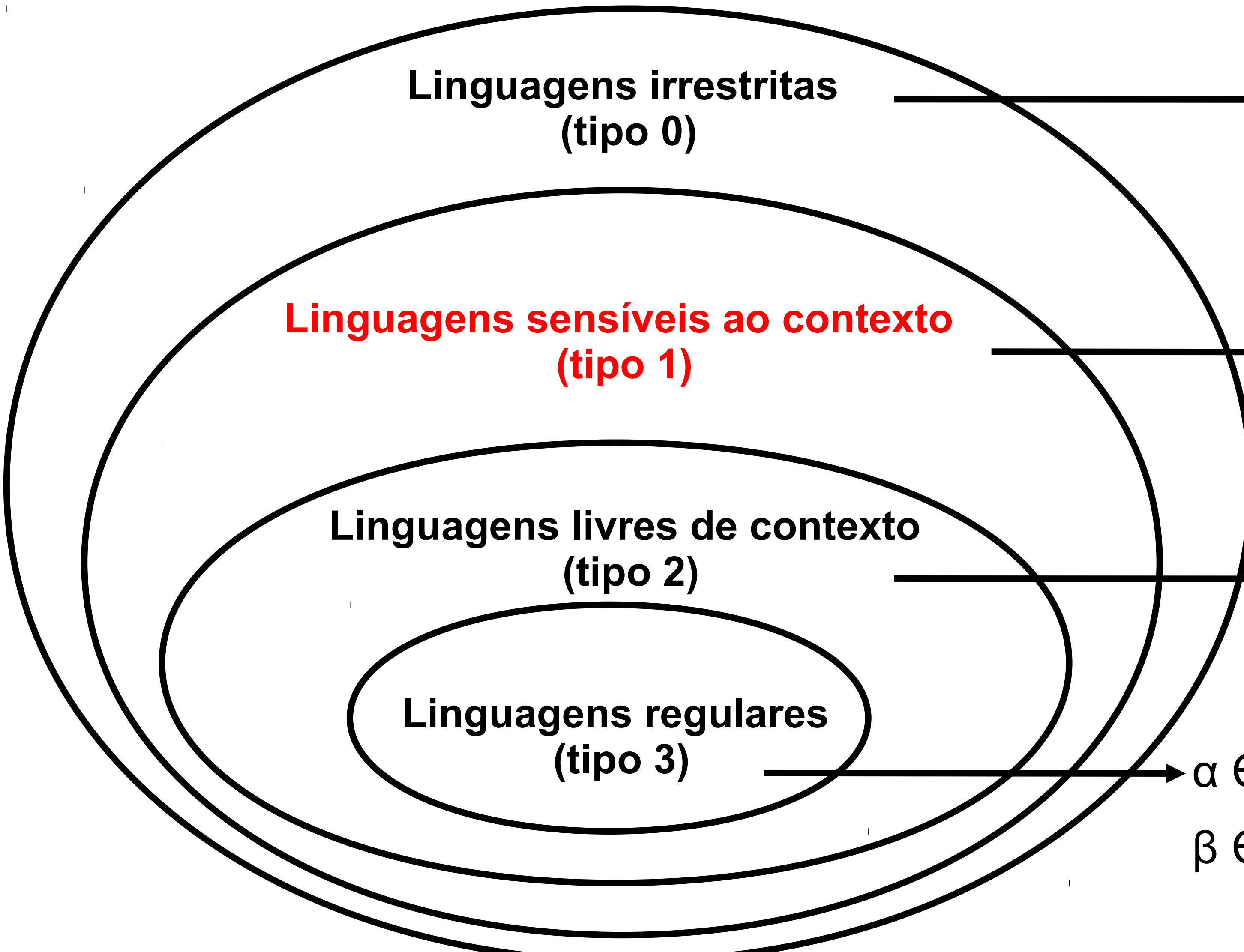
  Imprima as s_j que foram aceitas

Se w pertence a L, w é impressa um número infinito de vezes

Gramáticas Sensíveis ao Contexto

Hierarquia de Chomsky

$\alpha \rightarrow \beta$



$\alpha \in (V \cup \Sigma)^* V (V \cup \Sigma)^*$
$\beta \in (V \cup \Sigma)^*$

$\alpha \in (V \cup \Sigma)^* V (V \cup \Sigma)^*$
$\beta \in (V \cup \Sigma)^*$
$|\alpha| <= |\beta|$

$\alpha \in V$

$\beta \in (V \cup \Sigma)^*$

$\alpha \in V$

$\beta \in \Sigma_\varepsilon$, $\beta \in V$, $\beta \in (V\Sigma$ ou $\Sigma V)$

Exemplo 1

Que linguagem essa gramática gera?

$$
\begin{aligned}
P = \{\ S &\rightarrow aSBC \\
S &\rightarrow aBC, \\
CB &\rightarrow BC, \\
aB &\rightarrow ab, \\
bB &\rightarrow bb, \\
bC &\rightarrow bc, \\
cC &\rightarrow cc\}
\end{aligned}
$$

Exemplo 1

Que linguagem essa gramática gera?

$P = \{\ S \rightarrow aSBC$

$S \rightarrow aBC,$

$CB \rightarrow BC,$

$aB \rightarrow ab,$

$bB \rightarrow bb,$

$bC \rightarrow bc,$

$cC \rightarrow cc\}$

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

Exemplo 2

Escreva uma gramática para a linguagem

L = { w | w possui a mesma quantidade de a's, b'c e c's}

Exemplo 2

Escreva uma gramática para a linguagem

L = { w | w possui a mesma quantidade de a's, b'c e c's}

P = { $S \rightarrow ABCS$

$S \rightarrow ABC$,

$AB \rightarrow BA$,

$AC \rightarrow CA$,

$BA \rightarrow AB$,

$BC \rightarrow CB$,

$CA \rightarrow AC$,

$CB \rightarrow BC$,

$A \rightarrow a$,

$B \rightarrow b$,

$C \rightarrow c$ }

Gramáticas e Linguagens sensíveis ao contexto

- Gramáticas sensíveis ao contexto (GSC) são monotônicas: o comprimento das formas sentenciais durante a derivação de uma sentença nunca sofre redução
- Rigorosamente, uma linguagem L é sensível ao contexto se e somente se:
  - ε não pertence a L e L = L(G), onde G é GSC, ou
  - ε pertence a L e L – {ε} pode ser gerada por uma GSC

Gramáticas e Linguagens sensíveis ao contexto

- Se ε pertence a L, aceita-se colocar a regra $S \rightarrow \varepsilon$ se S for o símbolo inicial e S não aparecer do lado direito de nenhuma regra

- Uma linguagem é estritamente sensível ao contexto se ele não for livre de contexto

Máquinas de Turing com fita limitada

Definição semelhante à da Máquina de Turing

Diferença:

- Tamanho fita de trabalho = tamanho da entrada + 2
- Fita inicia e termina com símbolos delimitadores (por exemplo (“<” e “>”) não pertencentes ao alfabeto de fita nem ao alfabeto da linguagem

Exemplo

Adapte a Máquina de Turing que reconhece a linguagem $B = \{w\#w \mid w \text{ pertence a } \{0,1\}^* \}$.

Máquina de Turing com fita ilimitada