ACH2043 INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

Aula 17
Cap 4.1 – Linguagens Decidíveis

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

Cap 4 – Decidibilidade

Cap. 4.1 – Linguagens Decidíveis

Motivação

Saber se um problema é indecidível é importante para podermos fazer uma simplificação do mesmo a fim de propormos uma solução algorítmica.

Problemas decidíveis concernentes a linguagens regulares

- Problema da aceitação de uma cadeia w por um AFD B
- Como escrever esse problema em forma de uma linguagem?

Problemas decidíveis concernentes a linguagens regulares

- Problema da aceitação de uma cadeia w por um AFD B
- Como escrever esse problema em forma de uma linguagem?
- $A_{AFD} = \{<B,w> \mid B$ é um AFD que reconhece a cadeia w$\}$
- Mostrar que A_{AFD} é decidível é o mesmo que provar que o problema de aceitação é decidível

TEOREMA 4.1

A_{AFD} é uma linguagem decidível.

IDÉIA DA PROVA Simplesmente precisamos apresentar uma MT M que decide A_{AFD}.

$M =$ "Sobre a entrada $\langle B, w \rangle$, onde B é um AFD, e w, uma cadeia:

1. Simule B sobre a entrada w.
2. Se a simulação termina em um estado de aceitação, *aceite*. Se ela termina em um estado de não-aceitação, *rejeite*."

Prova (só alguns detalhes)

Primeiro, vamos examinar a entrada $\langle B, w \rangle$. Ela é uma representação de um AFD B juntamente com uma cadeia w. Uma representação razoável de B é simplesmente uma lista de seus cinco componentes, Q, Σ, δ, q_0 e F. Quando M recebe sua entrada, M primeiro determina se ela representa apropriadamente um AFD B e uma cadeia w. Se não, M rejeita.

Então, M realiza a simulação diretamente. Ela mantém registro do estado atual de B e da posição atual de B na entrada w escrevendo essa informação na sua fita. Inicialmente, o estado atual de B é q_0 e a posição atual de B sobre a entrada é o símbolo mais à esquerda de w. Os estados e a posição são atualizados conforme a função de transição especificada δ. Quando M termina de processar o último símbolo de w, M aceita a entrada se B estiver em um estado de aceitação; M rejeita a entrada se B estiver em um estado de não-aceitação.

Semelhantemente...

$A_{\mathsf{AFN}} = \{\langle B, w\rangle \mid B$ é um AFN que aceita a cadeia de entrada $w\}$.

Semelhantemente...

$A_{\mathsf{AFN}} = \{\langle B, w\rangle \mid B$ é um AFN que aceita a cadeia de entrada $w\}$.

TEOREMA 4.2

A_{AFN} é uma linguagem decidível.

PROVA

$N =$ "Sobre a entrada $\langle B, w\rangle$ onde B é um AFN, e w, uma cadeia:

1. Converta AFN B para um AFD equivalente C, usando o procedimento para essa conversão dado no Teorema 1.39.
2. Rode a MT M do Teorema 4.1 sobre a entrada $\langle C, w\rangle$
3. Se M aceita, *aceite*; caso contrário, *rejeite*."

Semelhantemente...

$A_{\mathsf{EXR}} = \{\langle R, w\rangle \mid R$ é uma expressão regular que gera a cadeia $w\}$.

Semelhantemente...

$$A_{\mathsf{EXR}} = \{\langle R, w\rangle \mid R \text{ é uma expressão regular que gera a cadeia } w\}.$$

TEOREMA 4.3 ....

A_{EXR} é uma linguagem decidível.

PROVA A seguinte MT P decide A_{EXR}.

P = "Sobre a entrada $\langle R, w\rangle$ onde R é uma expressão regular e w é uma cadeia:

1. Converta a expressão regular R para um AFN equivalente A usando o procedimento para essa conversão dado no Teorema 1.54.
2. Rode a MT N sobre a entrada $\langle A, w\rangle$.
3. Se N aceita, *aceite*; se N rejeita, *rejeite*."

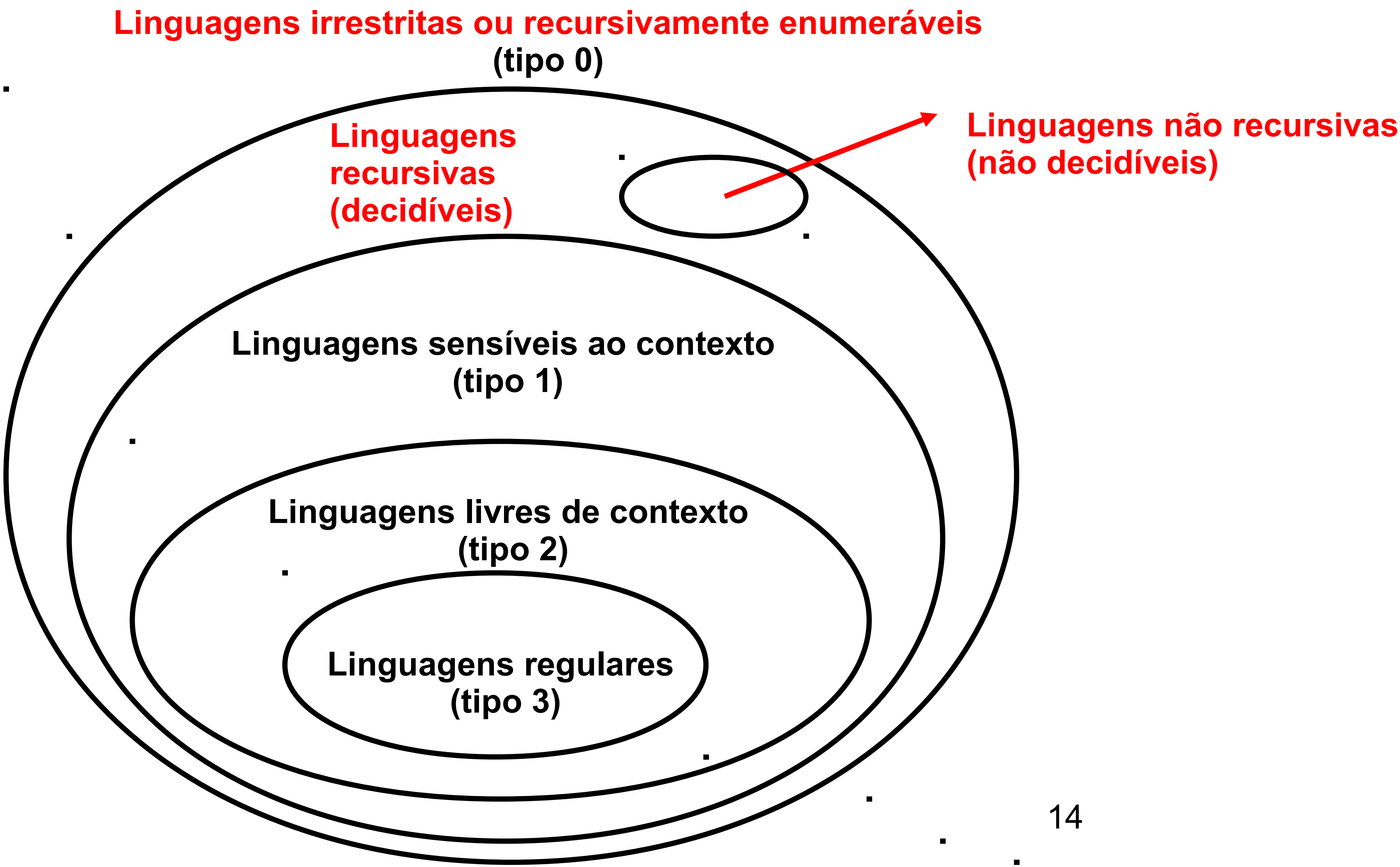
Semelhantemente...

- E para gramáticas regulares?

Problemas decidíveis concernentes a linguagens regulares

- Linguagens regulares são decidíveis (lembram da Hierarquia de Chomsky?)

Hierarquia de Chomsky



Teste de vacuidade

$$V_{\mathsf{AFD}} = \{\langle A\rangle \mid A \text{ é um } \mathsf{AFD} \text{ e } L(A) = \emptyset\}.$$

Teste de vacuidade

$$V_{\mathsf{AFD}} = \{\langle A \rangle \mid A \text{ é um } \mathsf{AFD} \text{ e } L(A) = \emptyset\}.$$

TEOREMA 4.4

V_{AFD} é uma linguagem decidível.

Teste de vacuidade

PROVA Um AFD aceita alguma cadeia sse é possível atingir um estado de aceitação a partir do estado inicial passando pelas setas do AFD. Para testar essa condição, podemos projetar uma MT T que usa um algoritmo de marcação similar àquele utilizado no Exemplo 3.23. (que decide se G é um grafo não-direcionado conexo)

T = "Sobre a entrada $\langle A \rangle$ onde A é um AFD:

1. Marque o estado inicial de A.
2. Repita até que nenhum estado novo venha a ser marcado:
3. Marque qualquer estado que tenha uma transição chegando nele a partir de qualquer estado que já está marcado.
4. Se nenhum estado de aceitação estiver marcado, *aceite*; caso contrário, *rejeite*."

Equivalência de dois AFDs

$$EQ_{\mathsf{AFD}} = \{\langle A, B\rangle \mid A \text{ e } B \text{ são AFDs e } L(A) = L(B)\}.$$

Equivalência de dois AFDs

$$EQ_{\mathsf{AFD}} = \{\langle A, B\rangle \mid A \text{ e } B \text{ são AFDs e } L(A) = L(B)\}.$$

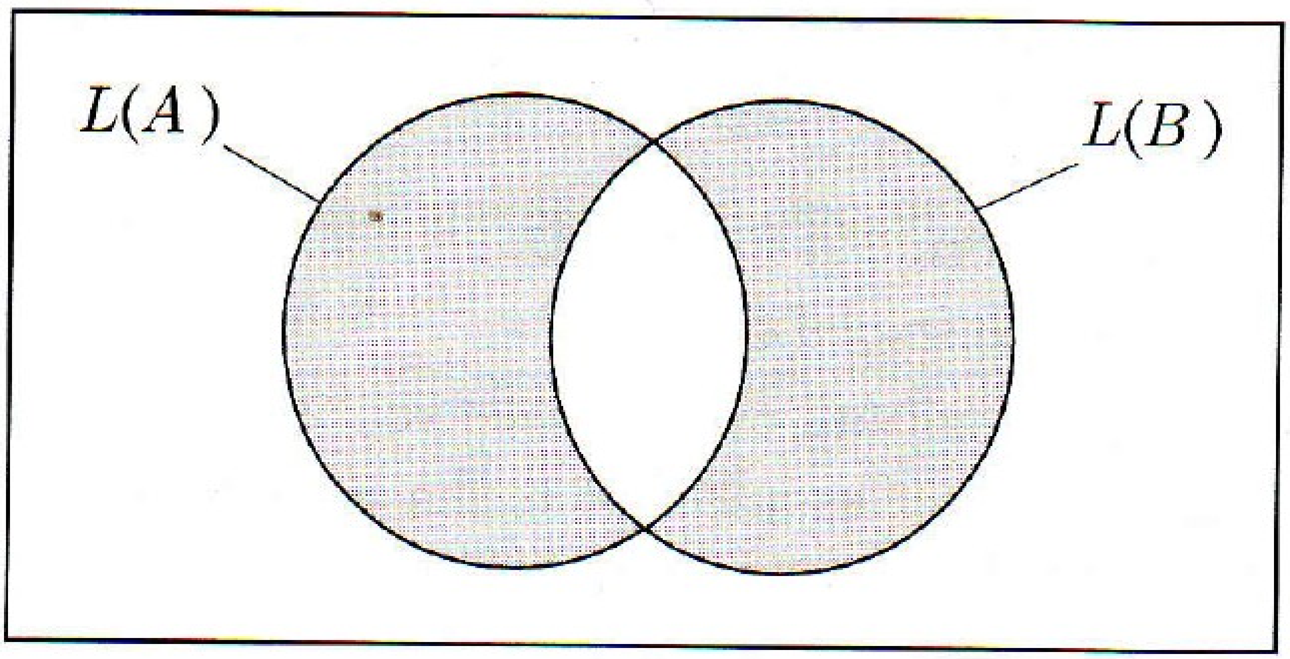
TEOREMA 4.5

EQ_{AFD} é uma linguagem decidível.

Equivalência de dois AFDs

PROVA Para provar esse teorema, usamos o Teorema 4.4. Construímos um novo AFD C a partir de A e B, tal que C aceita somente aquelas cadeias que são aceitas ou por A ou por B, mas não por ambos. Conseqüentemente, se A e B reconhecem a mesma linguagem, C não aceitará nada. A linguagem de C é

$$L(C) = \left(L(A) \cap \overline{L(B)}\right) \cup \left(\overline{L(A)} \cap L(B)\right).$$



Equivalência de dois AFDs

F = "Sobre a entrada $\langle A, B\rangle$, onde A e B são AFDs:

1. Construa o AFD C conforme descrito.
2. Rode a MT T do Teorema 4.4 sobre a entrada $\langle C\rangle$.
3. Se T aceita, *aceite*. Se T rejeita, *rejeite*."

O que pode ser feito com os algoritmos utilizados nas provas de fechamento das linguagens regulares com relação à complementação, união e intersecção (cap 1)

Problemas decidíveis concernentes a linguagens livres de contexto

$$A_{\mathsf{GLC}} = \{\langle G, w\rangle \mid G \text{ é uma } \mathsf{GLC} \text{ que gera a cadeia } w\}$$

Problemas decidíveis concernentes a linguagens livres de contexto

$$A_{\mathsf{GLC}} = \{\langle G, w\rangle \mid G \text{ é uma GLC que gera a cadeia } w\}$$

TEOREMA 4.7

A_{GLC} é uma linguagem decidível.

Problemas decidíveis concernentes a linguagens livres de contexto

IDÉIA DA PROVA Para a GLC G e a cadeia w, queremos determinar se G gera w. Uma idéia é usar G para passar por todas as derivações para determinar se alguma delas é uma derivação de w. Essa idéia não funciona, pois uma quantidade infinita de derivações pode ter que ser testada. Se G não gera w, esse algoritmo nunca pararia. Essa idéia leva a uma máquina de Turing que é um reconhecedor, mas não um decisor, para A_{GLC}.

Problemas decidíveis concernentes a linguagens livres de contexto

IDÉIA DA PROVA Para a GLC G e a cadeia w, queremos determinar se G gera w. Uma idéia é usar G para passar por todas as derivações para determinar se alguma delas é uma derivação de w. Essa idéia não funciona, pois uma quantidade infinita de derivações pode ter que ser testada. Se G não gera w, esse algoritmo nunca pararia. Essa idéia leva a uma máquina de Turing que é um reconhecedor, mas não um decisor, para A_{GLC}.

Para tornar essa máquina de Turing um decisor, precisamos garantir que o algoritmo tenta somente uma quantidade finita de derivações. No Problema 2.26 (página 136), mostramos que, se G estivesse na forma normal de Chomsky, qualquer derivação de w teria $2n - 1$ passos, onde n é o comprimento de w. Nesse caso, verificar apenas as derivações com $2n - 1$ passos para determinar se G gera w seria suficiente. Existe somente uma quantidade finita de tais derivações. Podemos converter G para a forma normal de Chomsky, usando o procedimento dado na Seção 2.1.

Problemas decidíveis concernentes a linguagens livres de contexto

PROVA A MT S para A_{GLC} segue.

$S =$ "Sobre a entrada $\langle G, w\rangle$, onde G é uma GLC, e w, uma cadeia:

1. Converta G para uma gramática equivalente na forma normal de Chomsky.
2. Liste todas as derivações com $2n - 1$ passos, onde n é o comprimento de w, exceto se $n = 0$; nesse último caso, liste todas as derivações com 1 passo.
3. Se alguma dessas derivações gera w, *aceite*; se não, *rejeite*."

Problemas decidíveis concernentes a linguagens livres de contexto

PROVA A MT S para A_{GLC} segue.

S = "Sobre a entrada $\langle G, w \rangle$, onde G é uma GLC, e w, uma cadeia:

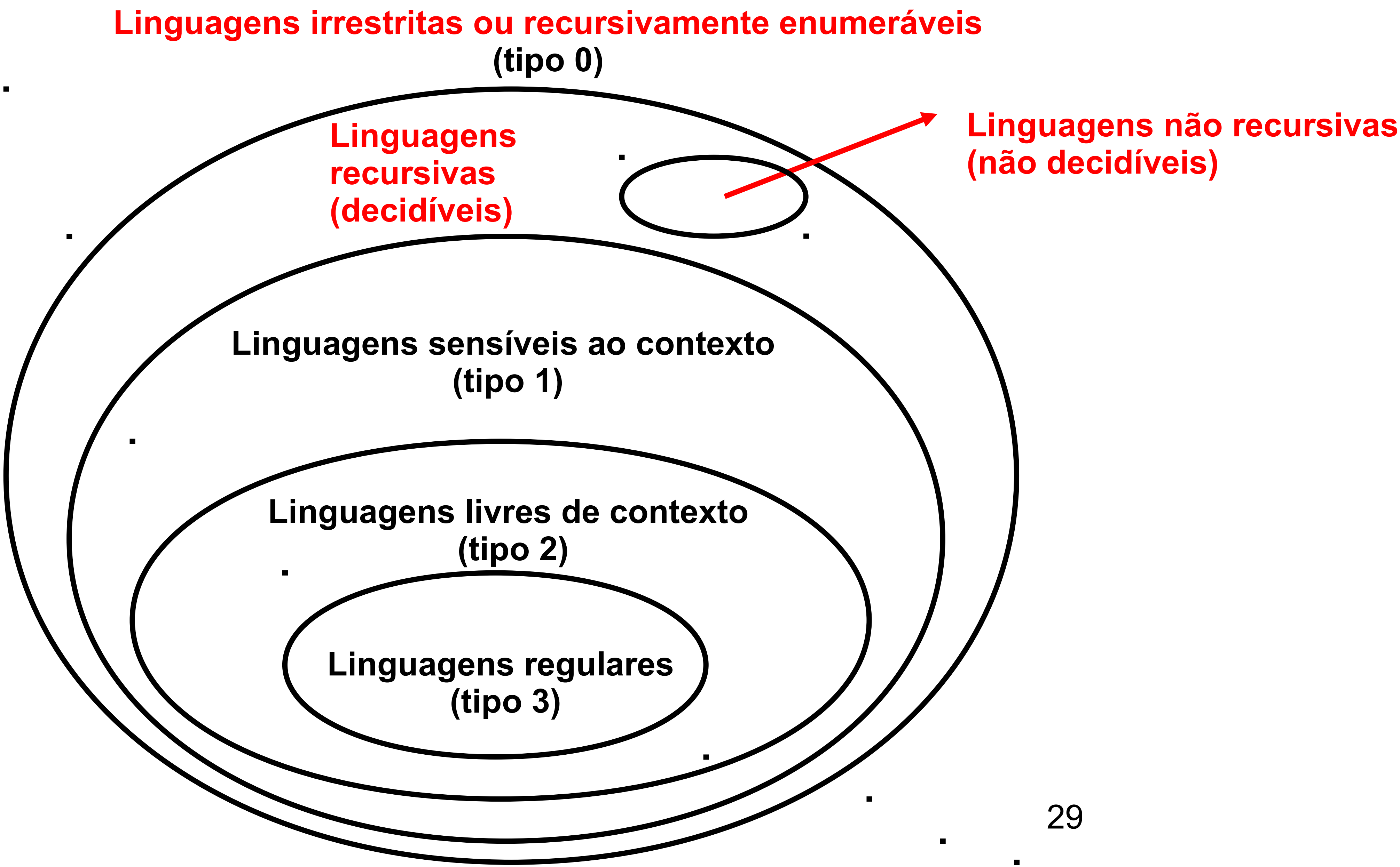
1. Converta G para uma gramática equivalente na forma normal de Chomsky.
2. Liste todas as derivações com $2n - 1$ passos, onde n é o comprimento de w, exceto se $n = 0$; nesse último caso, liste todas as derivações com 1 passo.
3. Se alguma dessas derivações gera w, *aceite*; se não, *rejeite*."

Ineficiente (exponencial em n), mas funciona!
Há um algoritmo mais eficiente para isso – CYK – que veremos depois (Teorema 7.16)

Problemas decidíveis concernentes a linguagens livres de contexto

- Linguagens livres de contexto são decidíveis (lembram da Hierarquia de Chomsky?)

Hierarquia de Chomsky



Problemas decidíveis concernentes a linguagens livres de contexto

- Linguagens livres de contexto são decidíveis (lembram da Hierarquia de Chomsky?)
- Mas como eu provo que GLCs são decidíveis?

Problemas decidíveis concernentes a linguagens livres de contexto

- Linguagens livres de contexto são decidíveis (lembram da Hierarquia de Chomsky?)
- Mas como eu provo que GLCs são decidíveis?

PROVA Seja G uma GLC para A e projetemos uma MT M_G que decide A. Construimos uma cópia de G dentro de M_G. Ela funciona da seguinte maneira.

$M_G =$ "Sobre a entrada w:

1. Rode a MT S sobre a entrada $\langle G, w \rangle$
2. Se essa máquina aceita, *aceite*; se ela rejeita, *rejeite*."

Problemas decidíveis concernentes a linguagens livres de contexto

- Linguagens livres de contexto são decidíveis (lembram da Hierarquia de Chomsky?)
- Mas como eu provo que GLCs são decidíveis?
- Poderia também utilizar autômatos a pilha (AP) na prova?
  – Má ideia... a dificuldade é que o AP pode ser não-determinístico, e podemos cair em um ramo da computação que não páre...

Vacuidade de GLCs

$$V_{\mathsf{GLC}} = \{\langle G\rangle |\ G \text{ é uma } \mathsf{GLC} \text{ e } L(G) = \emptyset\}.$$

Vacuidade de GLCs

$$V_{\mathrm{GLC}} = \{\langle G\rangle \mid G \text{ é uma GLC e } L(G) = \emptyset\}.$$

TEOREMA 4.8

V_{GLC} é uma linguagem decidível.

Vacuidade de GLCs

IDÉIA DA PROVA

- Testar se a GLC gera alguma cadeia de terminais => testar se a variável inicial gera uma cadeia de terminais
- Marcar cada variável que gera uma cadeia de terminais
  - Terminais
  - Variáveis que estão no lado esquerdo de pelo menos uma regra cujo lado direito está todo marcado
- Verificar se a variável inicial está marcada

Vacuidade de GLCs

PROVA

$R =$ "Sobre a entrada $\langle G \rangle$, onde G é uma GLC:

1. Marque todos os símbolos terminais em G.
2. Repita até que nenhuma variável venha a ser marcada:
3. Marque qualquer variável A onde G tem uma regra $A \to U_1 U_2 \cdots U_k$ e cada símbolo $U_1, \ldots, U_k$ já tenha sido marcado.
4. Se a variável inicial não está marcada, *aceite*; caso contrário, *rejeite*."

Equivalência de GLCs

$$EQ_{\mathsf{GLC}} = \{\langle G, H\rangle \mid G \text{ e } H \text{ são GLCs e } L(G) = L(H)\}.$$

Equivalência de GLCs

$$EQ_{\mathsf{GLC}} = \{\langle G, H\rangle \mid G \text{ e } H \text{ são GLCs e } L(G) = L(H)\}.$$

- Técnica semelhante a mostrar se dois AFDs são equivalentes?

Equivalência de GLCs

$$EQ_{\text{GLC}} = \{\langle G, H \rangle \mid G \text{ e } H \text{ são GLCs e } L(G) = L(H)\}.$$

- Técnica semelhante a mostrar se dois AFDs são equivalentes?
- Problema: LLCs não são fechadas com relação às operações de complementação e intersecção!

Equivalência de GLCs

$$EQ_{\mathsf{GLC}} = \{\langle G, H\rangle \mid G \text{ e } H \text{ são GLCs e } L(G) = L(H)\}.$$

- Técnica semelhante a mostrar se dois AFDs são equivalentes?
- Problema: LLCs não são fechadas com relação às operações de complementação e intersecção!
- Na verdade, EQ_{GLC} é indecidível!