Indecidibilidade e redução

Universidade Federal de Campina Grande – UFCG
Centro de Engenharia Elétrica e Informática – CEEI
Departamento de Sistemas e Computação – DSC
Professor: Andrey Brito
Período: 2023.2

Indecidibilidade

- Existem mesmo problemas indecidíveis?

- A_{MT} não é decidível
    - Dado um programa e uma entrada, simular aquela entrada
    - A_{MT} = {<M,w> | M é uma MT e M aceita w}
    - Mais algo, verificação de software: verificar propriedades interessantes de “programas” não é decidível

- O que faz a máquina que tem como linguagem A_{MT}?
    - Lembre do testador de A_{AFD}

Outro problema indecidível

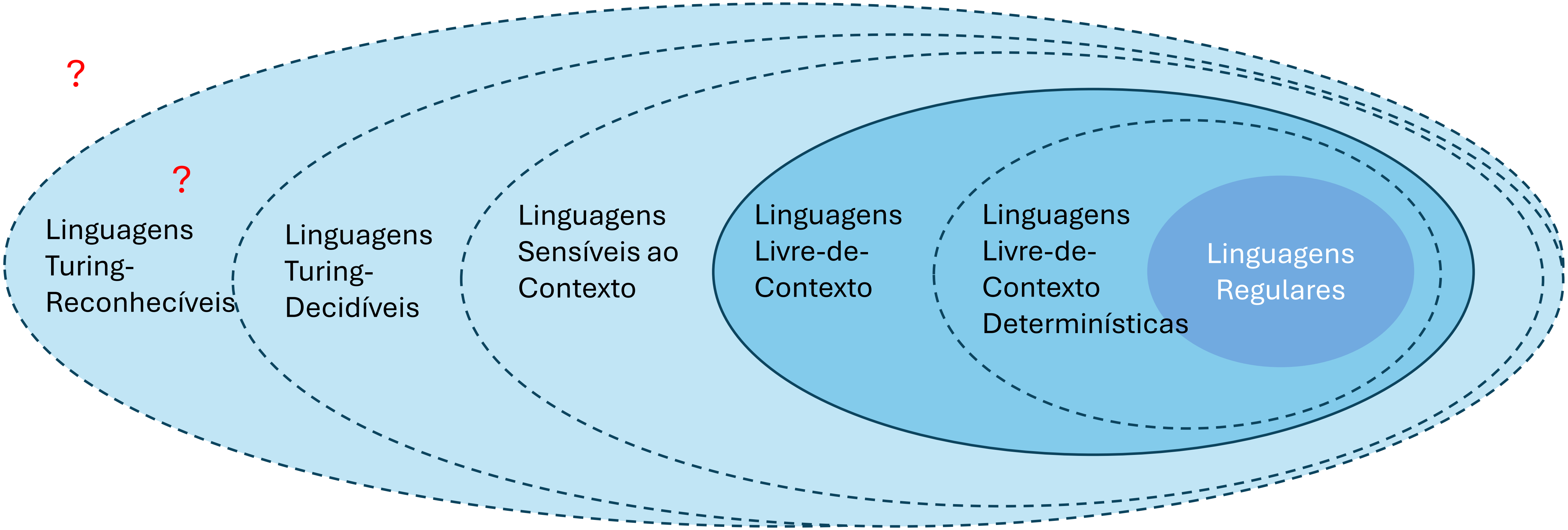
- PCP: Problema de correspondência de Post
  - Dado um conjunto de dominós

$$\left\{ \begin{pmatrix} a \\ a\,b \end{pmatrix} \begin{pmatrix} c\,a \\ a\,b \end{pmatrix} \begin{pmatrix} b \\ c \end{pmatrix} \begin{pmatrix} b\,d \\ d \end{pmatrix} \right\}$$

  - Verificar se existe uma sequência, onde a cadeia de cima é igual à de baixo
  - (Pode repetir peças e não precisa usar tudo)

$$\begin{pmatrix} a \\ a\,b \end{pmatrix} \begin{pmatrix} b\,d \\ d \end{pmatrix} \qquad \begin{pmatrix} a \\ a\,b \end{pmatrix} \begin{pmatrix} b \\ c \end{pmatrix} \begin{pmatrix} c\,a \\ a\,b \end{pmatrix} \begin{pmatrix} b\,d \\ d \end{pmatrix}$$

Hierarquia de linguagens

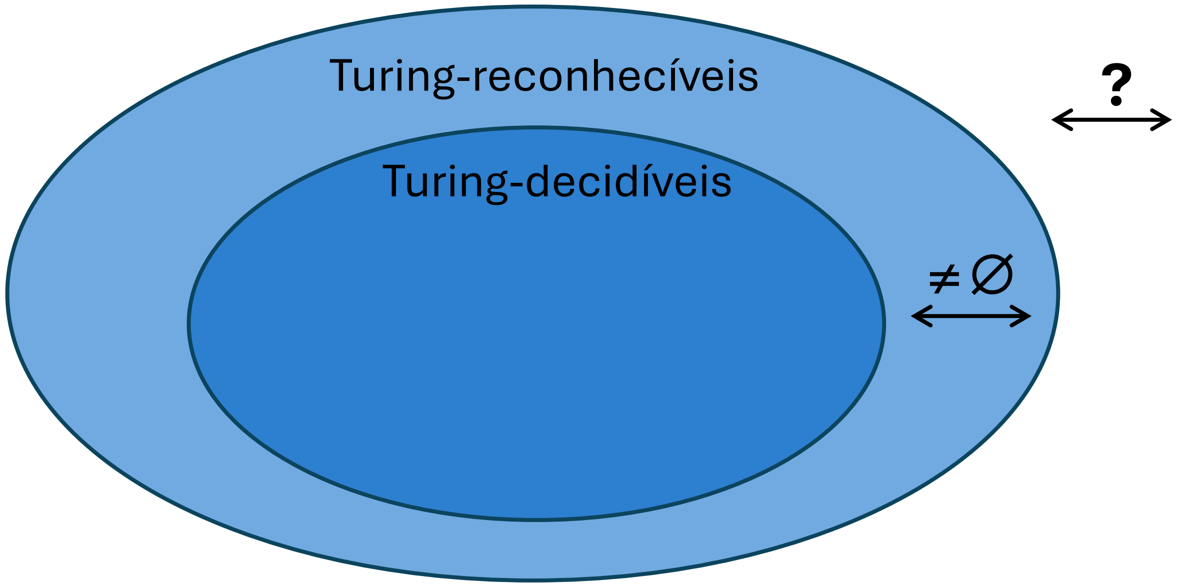


Existe algo fora de LTR?

Sobre o problema da parada e H...

- Não necessariamente H simula a máquina
  - Poderia fazer uma “análise”
  - Mesmo assim H não pode existir

- A_{MT} não é decidível, mas é reconhecível...



Existe algo fora de LTR?

- Se eu tiver duas linguagens L e seu complemento, /L
  - Sei que elas são Turing-reconhecíveis
  - Eu posso dizer algo sobre decidibilidade delas?

- Consequências
  - LTR não é fechada por complemento
  - Existe algo fora de LTR, as linguagens Turing-irreconhecíveis

Indecidibilidade

- Teorema: uma linguagem é decidível se e somente se, tanto ela quanto o complemento forem reconhecíveis
  - Co-Turing reconhecível: complemento reconhecível

- Prova:
  - Se A é decidível, A e $\bar{A}$ são reconhecíveis
  - Se A e $\bar{A}$ são reconhecíveis, A é decidível

- M = “Com entrada w:
  - Simule as M_Ae $M_{/A}$ em paralelo (ex., uma quantidade limitada de tempo em cada uma e que cresce a cada tentativa)
  - Se M_A aceita, aceite, se $M_{/A}$ aceita, rejeite.”

Turing-irreconhecível

- Mas então se A_{MT} não é decidível, mas é reconhecível: $\overline{A_{MT}}$ não pode ser sequer reconhecível

- Mas quem seria $\overline{A_{MT}}$?
  - A_{MT} = {<M,w> | M é uma MT e M aceita w}
  - $\overline{A_{MT}}$ = {<M,w> | M é uma MT e M não aceita w}

Provas de indecidibilidade

- Como mostrar que um problema (ou a linguagem que o representa) é indecidível?
  - Matematicamente (A_{MT}, existência de raízes inteiras para um polinômio com múltiplas variáveis)
  - O complemento da linguagem em questão não é reconhecível
  - Redução: ele poderia ser usado para resolver um problema que é indecidível

O verdadeiro problema da parada: $HALT_{TM}$ ($PARADA_{MT}$)

- $HALT_{TM}$ só se preocupa em determinar se a máquina/programa a ser testado para ou não com aquela entrada

O verdadeiro problema da parada: $HALT_{TM}$ ($PARADA_{MT}$) é indecidível

- Ideia: $HALT_{TM}$ não pode ser decidível
  - Se $HALT_{TM}$ fosse decidível, seriam simples construir um decididor para A_{TM}
  - No entanto, provamos que A_{TM} não é decidível

- Se R decide $HALT_{TM}$ (aceita = para), S decide A_{MT}

- Quem seria o programa/máquina S?

O verdadeiro problema da parada: $HALT_{TM}$ ($PARADA_{MT}$) é indecidível

- Ideia: $HALT_{TM}$ não pode ser decidível
  - Se $HALT_{TM}$ fosse decidível, seriam simples construir um decididor para A_{TM}
  - No entanto, provamos que A_{TM} não é decidível

- Se R decide $HALT_{TM}$ (aceita = para), S decide A_{MT}

- S = “com entrada <M, w>, onde M é uma descrição de uma MT e w uma palavra:
  - Execute R com a entrada <M, w>
  - Se R rejeita, rejeite, caso contrário, simule M com a entrada w.
  - Se M aceita w, aceite. Se M rejeita, rejeite.”

$VAZIO_{MT}$ é indecidível?

- $VAZIO_{MT}$ = { <M> | M é uma MT e L(M) = $\varnothing$}

- Novamente, a ideia é mostrar que se $VAZIO_{MT}$ fosse decidível (existindo uma MT R), conseguiríamos usar isso para decidir A_{MT} (MT S)
  - Como não podemos decidir $A_{MT,}$ logo $VAZIO_{MT}$ não pode ser decidível

- Possibilidade 1: rodar R com <M> e ver se aceita
  - Se R aceitar, L(M) = $\varnothing$ e portanto M não aceita w
  - Mas e se rejeitar? Tudo que saber é que L(M) $\neq \varnothing$
  - Ou seja, eu não consigo resolver S usando R, bom sinal?!?

$VAZIO_{MT}$ é indecidível

- Possibilidade 2: crie uma variante de M, M_1
    - Que rejeite qualquer entrada diferente de *w* imediatamente
    - Se a entrada for w, então ela prossegue com o código original de M

- O que M_1 aceita?

$VAZIO_{MT}$ é indecidível

- Possibilidade 2: crie uma variante de M, M_1
  - Que rejeite qualquer entrada diferente de *w* imediatamente
  - Se a entrada for w, então ela prossegue com o código original de M

- O que M_1 aceita?
  - M_1 aceita ou aquela palavra *w* ou nada
  - Usamos então R para verificar qual das duas situações ocorre

V_{MT} é indecidível?

- V_{MT} = { <M> | M é uma MT que para com entrada λ}

V_{MT} é indecidível

- V_{MT} = { <M> | M é uma MT que para com entrada λ}

- Seja uma MT R que decide V_{MT}

- S = "com entrada <M, *w*>, onde M é uma descrição de uma MT e *w* uma cadeia:
  - Construir uma variante de M, M_1:
    - M_1 = "Ignorando a entrada:
      - Escreve *w* na fita.
      - Simula M com entrada <M, w>, aceita se M aceitaria e rejeita se ela rejeitaria."
  - Executa R com entrada <M_1>
  - Se R aceita, aceite. Caso contrário, rejeite."

- Se R aceitar, é por que M_1 aceita a palavra vazia... mas isso só acontece se M aceita w

Redução

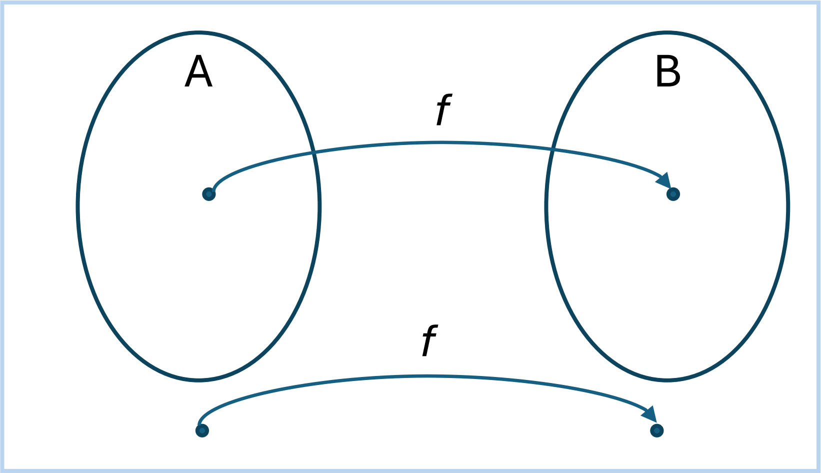
- Transformar a solução de um problema A em uma solução para o problema B
  - A é redutível a B se uma solução de B pode ser usada para resolver A
  - “Resolver A se reduz a resolver B”

- Uma estratégia para avaliar decidibilidade
  - Se A é redutível a B e B é decidível, A é decidível
  - Se A é redutível a B, mas A é indecidível, B é indecidível
  - Se A é redutível a B e B é indecidível… ?

- B pode ser “aparentemente” mais fácil que A

Redução por mapeamento

- A linguagem A é redutível por mapeamento à linguagem B ($A \leq_m B$) se existe uma função computável $f : \Sigma^* \to \Sigma^*$ tal que:
  - $w \in A$ se e somente se $f(w) \in B$
  - A função f é chamada de redução



Redução por mapeamento (intuição!)

- Os problemas
  - B = { x | x é par}
  - A = { x | x é ímpar}

- Lembre: *w* $\in$ A se e somente se *f(w)* $\in$ B

- Será que existe um f(w)?
  - Uma função computável...
  - ... e que me garante que se y $\in$ A, então f(y) $\in$ B??

Redução por mapeamento (só intuição!)

- Os problemas
  - B = { x | x é par}
  - A = { x | x é ímpar}

- Lembre: $w \in A$ se e somente se $f(w) \in B$

- Será que existe um f(w)?
  - Uma função computável...
  - ... e que me garante que se $y \in A$, então $f(y) \in B$??
  - Que tal $f(w) = w + 1$?

Redução por mapeamento

- Os problemas
  - B = { x | x é par}
  - A = { x | x é ímpar}

- Com *f(w) = w + 1* sabemos que A é redutível a B por mapeamento pois:
  - Dado MT B que me diz se uma entra w é par
  - Eu posso construir uma MT para A:
    - Pegue a entrada w e use uma calculadora que implemente f(w)
    - Chame a MT B, aceite se B aceitar, rejeite se B rejeitar

Decidibilidade

- Se A $\leq_m$ B e B é decidível, A é decidível
- Seja M uma MT que decide B e *f* uma função de redução
- Prova: N =“Com entrada w:
  - Compute f(w).
  - Execute M com entrada f(w), se M aceita, aceite. Se M rejeita, rejeite.”
- O formato da máquina N, que decide A, tem que ser sempre esse!

Indecidibilidade de PARADA_{MT}

- Prova alternativa usando mapeamento, ou seja, existe uma função f computada pela seguinte MT...

- F = "Dado uma entrada <M, w>, onde M é uma descrição de MT:
  - Construa a seguinte MT M': "Com entrada x...
    - Execute M com entrada x.
    - Se M aceita, aceite.
    - Se M rejeita, entre em loop."
  - Grave <M', w> na fita e limpe o resto."

- <M, w> $\in A_{TM}$ se e somente se <M', w> $\in \text{PARADA}_{MT}$

Outro exemplo

- $VAZIO_{MT} \leq_m IGUAL_{MT}$
  - $VAZIO_{MT}$ (que sabemos ser indecidível) verifica se para entrada M, $L(M)=\emptyset$
  - $IGUAL_{TM}$ (a ser testado) testa se duas MTs têm a mesma linguagem

Outro exemplo

- $VAZIO_{MT} \leq_m IGUAL_{MT}$
  - $VAZIO_{MT}$ (que sabemos ser indecidível) verifica se para entrada M, $L(M)=\emptyset$
  - $IGUAL_{TM}$ (a ser testado) testa se duas MTs têm a mesma linguagem
  - Função de redução: transforme a entrada <M> em <M, M_1>, onde M_1 é uma descrição de uma máquina que rejeita tudo.
  - Bastaria executar a MT que decide $IGUAL_{TM}$ com <M, M_1>

Hierarquia de linguagens