20) DECIDIBILIDADE

Decidibilidade

→ Como qualquer ferramenta, MTs têm capacidades e limitações, que devem ser reconhecidas caso se queira usá-las bem.

→ Saber se algo é insolúvel pode salvar tempo mas também te dar perspectiva sobre computação.

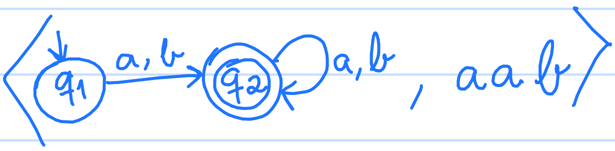
→ Já vimos vários problemas decidíveis:
- Dado um grafo, ele é conexo?
- Dados um conj. de objetos, existem dois iguais?
- Um dado número é potência de dois?
- Dados i, j, K, $i \times j = K$?

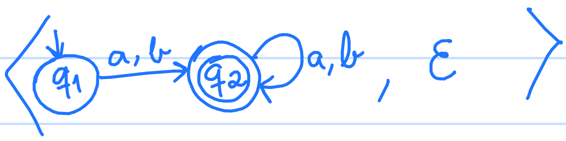
→ E alguns que não são:
- Um polinômio sobre múltiplas variáveis tem raiz inteira?

Exemplos

PROB: Dados um AFD e uma cadeia, ele a aceita?

EQUIV: $A_{AFD} = \{\langle A, w\rangle : A$ é AFD que aceita $w\}$





$((q_1, q_2), (a, b), ((q_1, a, q_2), (q_1, b, q_2), (q_2, a, q_2), (q_2, b, q_2)), q_1, (q_2))$

$M_{A_{AFD}}$ = "Sobre a entrada $\langle A, w\rangle$, onde A é AFD e w é cadeia:
(1) Simule A sobre w.
(2) Se a simulação termina em um estado final, aceite. Caso contrário, rejeite."

IMPORTANTE entender por quê podemos simular um AFD!

Exemplos

PROB: Dados um AFN e uma cadeia, ele a aceita?
EQUIV: $A_{AFN} = \{\langle A,w \rangle : A$ é AFN que aceita $w\}$

$M_{A_{AFN}}$ = "Sobre a entrada $\langle A,w \rangle$, onde A é AFN e w é cadeia:
(1) Converta A em um AFD B equivalente. *
(2) Execute $M_{A_{AFD}}$ sobre $\langle B,w \rangle$.
(3) Se $M_{A_{AFD}}$ aceitar, aceite.
Caso contrário, rejeite."

PROB: Dados uma ER e uma cadeia, ela a descreve?
EQUIV: $A_{ER} = \{\langle R,w \rangle : R$ é uma ER que descreve $w\}$.

$M_{A_{ER}}$ = "Sobre a entrada $\langle R,w \rangle$, onde R é ER e w é cadeia:
(1) Converta R em um AFN B equivalente.
(2) Execute $M_{A_{AFN}}$ sobre $\langle B,w \rangle$.
(3) Se $M_{A_{AFN}}$ aceitar, aceite.
Caso contrário, rejeite."

Por isso, qualquer coisa que dissermos sobre decidibilidade de problemas com relação a AFDs se aplica igualmente a AFNs e ERs.

PROB: Dado um AFD, ele aceita alguma cadeia?
EQUIV: ~~$V_{AFD} = \{\langle A \rangle : A$ é um AFD e $L(A) = \emptyset\}$~~
$V'_{AFD} = \{\langle A \rangle : A$ é um AFD e $L(A) \neq \emptyset\}$

$M_{V'_{AFD}}$ = "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 seja marcado:
Marque qualquer estado que tenha uma transição para ele saindo de um marcado.
(3) Se nenhum estado final estiver marcado, ~~aceite~~ rejeitar.
Caso contrário, ~~rejeite~~ aceitar."

CUIDADO: $V'_{AFD} \neq \overline{V_{AFD}}$

$\overline{V_{AFD}} = \{w : w$ não é da forma $\langle A \rangle$, onde A é um AFD$\} \cup V'_{AFD}$

PROB: Dados dois AFDs, eles são equivalentes?
EQUIV: $E_{AFD} = \{\langle A,B \rangle \mid A$ e B são AFDs e $L(A) = L(B)\}$

$M_{E_{AFD}}$ = "Sobre a entrada $\langle A,B \rangle$, onde A e B são AFDs:
(1) Construa um AFD C tal que $L(C) = (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$
(2) Execute $M_{V_{AFD}}$ sobre $\langle C \rangle$.
(3) Se $M_{V_{AFD}}$ aceitar, aceite.
Caso contrário, rejeite."

(note que $L(C) = \emptyset$ sse $L(A) = L(B)$)

(Como construir C?)

PROB: Dados uma GLC e uma cadeia, ela a gera?
EQUIV: $A_{GLC} = \{\langle G,w \rangle : G$ é GLC que gera $w\}$

T = "Sobre a entrada $\langle G,w \rangle$, onde G é GLC e w é cadeia:
(1) Para cada derivação de G:
Se ela gera w, aceite.
(2) Rejeite"

$\hookrightarrow$ Reconhecedora!!!

($S \to aS \mid \varepsilon$ tem nº infinito de derivações)

Forma normal de Chomsky: regras são da forma $A \to a$ (ou ε) ou $A \to BC$. Gramáticas assim derivam cadeias w com $|w| = n$ em $2n-1$ passos. Qualquer LLC pode ser gerada por uma GLC na FNC.

$M_{A_{GLC}}$ = "Sobre a entrada $\langle G,w \rangle$, onde G é GLC e w é cadeia:
(1) Converta G em uma GLC G' equivalente na FNC.
(2) Se $|w| = 0$, faça $m = 1$. Caso contrário, faça $m = 2|w| - 1$.
(3) Para cada derivação com m passos:
Se ela gera w, aceite.
(4) Rejeite."

Também por causa dos algoritmos de conversão entre GLCs e APs, qualquer resultado de decidibilidade que dissermos sobre GLCs vale igualmente para APs.

PROB: Dada uma GLC, ela gera alguma cadeia?

EQUIV: $V_{GLC} = \{ \langle G \rangle \mid G \text{ é GLC e } L(G) = \emptyset \}$

$S \to ABS$
$A \to ABCD \mid BA$
$B \to aAA$
$C \to bb$
$D \to DA \mid C$

$S \to ABS \mid C$
$A \to B \mid C$
$B \to aa$
$C \to b \mid A$

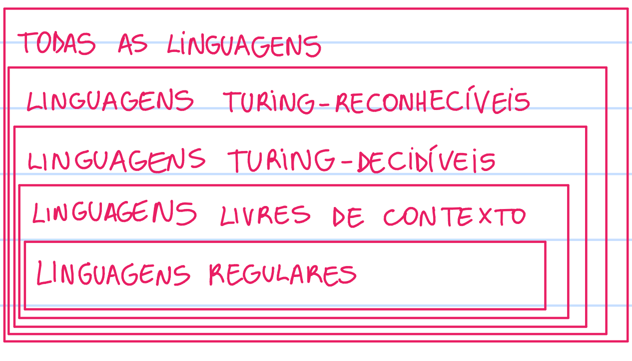
$M_{V_{GLC}}$ = "Sobre a entrada $\langle G \rangle$, onde G é uma GLC:
(1) Marque todos os terminais e o ε, se houver
(2) Repita até que nenhuma variável seja marcada:
Marque uma variável A tal que existe uma regra $A \to \alpha$ em que todo símbolo de α esteja marcado.
(3) Se a variável inicial está marcada, rejeite.
Caso contrário, aceite."

Relação entre LLCs e ling. Turing-decidíveis

TEOREMA: Toda linguagem livre de contexto é Turing-Decidível.

DEMONSTRAÇÃO: Seja A uma LLC. Então existe uma GLC G que a reconhece. A MT a seguir é decisora para A:

M_A = "Sobre a entrada w:
(1) Execute $M_{A_{GLC}}$ sobre $\langle G, w \rangle$
(2) Se $M_{A_{GLC}}$ aceitar, aceite.
Caso contrário, rejeite."



21) O PROBLEMA DA PARADA

Paradoxos

(1) Em uma cidade, existe apenas um barbeiro, do sexo masculino. Todo homem deve se manter barbeado, seja indo ao barbeiro ou fazendo ele mesmo.
O barbeiro só faz a barba daqueles que não se barbeiam
Quem barbeia o barbeiro?

(2) A frase seguinte é verdadeira.
A frase anterior é falsa.

Uma máquina que recebe máquinas

PROB: Dados uma MT e uma cadeia, ela a aceita?
EQUIV: $A_{MT} = \{ \langle M, w \rangle : M \text{ é MT que aceita } w \}$

Para mostrar que A_{MT} é Turing-decidível, precisamos construir uma MT que quando recebe $\langle M, w \rangle$ qualquer, decide se $\langle M, w \rangle \in A_{MT}$ ou se $\langle M, w \rangle \notin A_{MT}$.

O problema da Parada

PROB: Dados uma MT e uma cadeia, ela a aceita?
EQUIV: $A_{MT} = \{ \langle M, w \rangle : M \text{ é MT que aceita } w \}$

U = "Sobre a entrada $\langle M, w \rangle$: ⇒ Reconhecedora!!!
(1) Simule M sobre w.
(2) Se M aceita, aceite. Se M rejeita, rejeite."

U é uma máquina universal: ela simula outra MT.