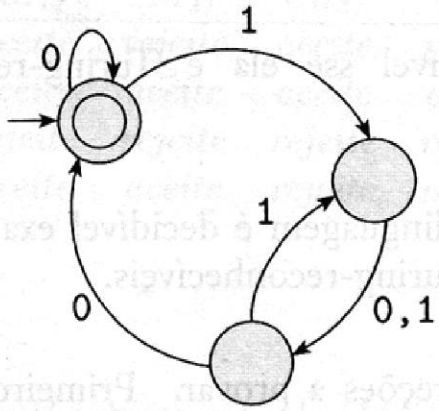
Introdução à Teoria da Computação Exercícios

Livro: Michel Sipser, Introdução à Teoria da Computação – 2ª Ed. – Capítulo 04

EXERCÍCIOS

[R]4.1 Responda a cada um dos itens abaixo para o AFD M e dê razões para suas respostas.



a. $\langle M, 0100\rangle \in A_{\text{AFD}}$?
b. $\langle M, 011\rangle \in A_{\text{AFD}}$?
c. $\langle M\rangle \in A_{\text{AFD}}$?
d. $\langle M, 0100\rangle \in A_{\text{EXR}}$?
e. $\langle M\rangle \in V_{\text{AFD}}$?
f. $\langle M, M\rangle \in EQ_{\text{AFD}}$?

4.2 Considere o problema de se determinar se um AFD e uma expressão regular são equivalentes. Expresse esse problema como uma linguagem e mostre que ele é decidível.

4.3 Seja $TODAS_{\text{AFD}} = \{\langle A\rangle \mid A \text{ é um AFD e } L(A) = \Sigma^*\}$. Mostre que $TODAS_{\text{AFD}}$ é decidível.

4.4 Seja $A\varepsilon_{\text{GLC}} = \{\langle G\rangle \mid G \text{ é uma GLC que gera } \varepsilon\}$. Mostre que $A\varepsilon_{\text{GLC}}$ é decidível.

4.5 Seja X o conjunto $\{1, 2, 3, 4, 5\}$ e Y o conjunto $\{6, 7, 8, 9, 10\}$. Descrevemos as funções $f: X \longrightarrow Y$ e $g: X \longrightarrow Y$ nas tabelas abaixo. Responda a cada item e dê uma razão para cada resposta negativa.

n	$f(n)$
1	6
2	7
3	6
4	7
5	6

n	$g(n)$
1	10
2	9
3	8
4	7
5	6

[R]**a.** f é um-para-um?
b. f é sobrejetora?
c. f é uma correspondência?
[R]**d.** g é um-para-um?
e. g é sobrejetora?
f. g é uma correspondência?

4.6 Seja $\mathcal{B}$ o conjunto de todas as seqüências infinitas sobre $\{0,1\}$. Mostre que $\mathcal{B}$ é incontável, usando uma prova por diagonalização.

4.7 Seja $T = \{(i,j,k) | \ i,j,k \in \mathcal{N}\}$. Mostre que T é contável.

4.8 Revise a maneira pela qual definimos conjuntos como sendo do mesmo tamanho na Definição 4.12 (página 184). Mostre que "é do mesmo tamanho" é uma relação de equivalência.

PROBLEMAS

[R]**4.9** Seja $INFINITA_{\text{AFD}} = \{\langle A \rangle | \ A \text{ é um AFD e } L(A) \text{ é uma linguagem infinita}\}$. Mostre que $INFINITA_{\text{AFD}}$ é decidível.

4.10 Seja $INFINITA_{\text{AP}} = \{\langle M \rangle | \ M \text{ é um AP e } L(M) \text{ é uma linguagem infinita}\}$. Mostre que $INFINITA_{\text{AP}}$ é decidível.

[R]**4.11** Seja $A = \{\langle M \rangle | \ M \text{ é um AFD que não aceita nenhuma cadeia contendo um número ímpar de 1s}\}$. Mostre que A é decidível.

4.12 Seja $A = \{\langle R,S \rangle | \ R \text{ e } S \text{ são expressões regulares e } L(R) \subseteq L(S)\}$. Mostre que A é decidível.

[R]**4.13** Seja $\Sigma = \{0,1\}$. Mostre que o problema de se determinar se uma GLC gera alguma cadeia em 1^* é decidível. Em outras palavras, mostre que

$$\{\langle G \rangle | \ G \text{ é uma GLC sobre } \{0,1\} \text{ e } 1^* \cap L(G) \neq \emptyset\}$$

é uma linguagem decidível.

[*]**4.14** Mostre que o problema de se determinar se uma GLC gera todas as cadeias em 1^* é decidível. Em outras palavras, mostre que $\{\langle G \rangle | \ G \text{ é uma GLC sobre } \{0,1\} \text{ e } 1^* \subseteq L(G)\}$ é uma linguagem decidível.

4.15 Seja $A = \{\langle R \rangle | \ R \text{ é uma expressão regular que descreve uma linguagem contendo pelo menos uma cadeia } w \text{ que tem } 111 \text{ como uma subcadeia (isto é, } w = x111y \text{ para alguma } x \text{ e alguma } y)\}$. Mostre que A é decidível.

4.16 Prove que EQ_{AFD} é decidível testando os dois AFDs sobre todas as cadeias até um certo tamanho. Calcule um tamanho que funcione.

[*]**4.17** Seja C uma linguagem. Prove que C é Turing-reconhecível sse existe uma linguagem decidível D tal que $C = \{x | \ \exists y \ (\langle x,y \rangle \in D)\}$.

4.18 Sejam A e B duas linguagens disjuntas. Digamos que a linguagem C ***separa*** A e B se $A \subseteq C$ e $B \subseteq \overline{C}$. Mostre que quaisquer duas linguagens co-Turing-reconhecíveis disjuntas são separáveis por alguma linguagem decidível.

4.19 Seja $S = \{\langle M \rangle | \ M \text{ é um AFD que aceita } w^{\mathcal{R}} \text{ sempre que ele aceita } w\}$. Mostre que S é decidível.

4.20 Uma linguagem é ***livre-de-prefixo*** se nenhum membro é um prefixo próprio de um outro membro. Seja $LIVRE\text{-}DE\text{-}PREFIXO_{\mathsf{EXR}} = \{R \mid R$ é uma expressão regular onde $L(R)$ é livre-de-prefixo$\}$. Mostre que $LIVRE\text{-}DE\text{-}PREFIXO_{\mathsf{EXR}}$ é decidível. Por que uma abordagem similar falha em mostrar que $LIVRE\text{-}DE\text{-}PREFIXO_{\mathsf{GLC}}$ é decidível?

R* **4.21** Digamos que um AFN é ***ambíguo*** se ele aceita alguma cadeia ao longo de dois ramos diferentes da computação. Seja $AMBIG_{\mathsf{AFN}} = \{\langle N \rangle \mid N$ é um AFN ambíguo$\}$. Mostre que $AMBIG_{\mathsf{AFN}}$ é decidível. (Sugestão: Uma maneira elegante de resolver este problema é construir um AFD apropriado e aí então rodar E_{AFD} sobre ele.)

4.22 Um ***estado inútil*** em um autômato com pilha nunca é atingido sobre qualquer cadeia de entrada. Considere o problema de se determinar se um autômato com pilha tem quaisquer estados inúteis. Formule esse problema como uma linguagem e mostre que ele é decidível.

R* **4.23** Seja $BAL_{\mathsf{AFD}} = \{\langle M \rangle \mid M$ é um AFD que aceita alguma cadeia contendo igual número de 0s e 1s$\}$. Mostre que BAL_{AFD} é decidível. (Dica: Os teoremas sobre LLCs são úteis aqui.)

* **4.24** Seja $PAL_{\mathsf{AFD}} = \{\langle M \rangle \mid M$ é um AFD que aceita algum palíndromo$\}$. Mostre que PAL_{AFD} é decidível. (Dica: Os teoremas sobre LLCs são úteis aqui.)

* **4.25** Seja $E = \{\langle M \rangle \mid M$ é um AFD que aceita alguma cadeia com mais 1s que 0s$\}$. Mostre que E é decidível. (Dica: Os teoremas sobre LLCs são úteis aqui.)

4.26 Seja $C = \{\langle G, x \rangle \mid G$ é uma GLC que gera alguma cadeia w, onde x é uma subcadeia de $w\}$. Mostre que C é decidível. (Sugestão: Uma solução elegante para esse problema usa o decisor para V_{GLC}.)

4.27 Seja $C_{\mathsf{GLC}} = \{\langle G, k \rangle \mid L(G)$ contém exatamente k cadeias, onde $k \geq 0$, ou então $k = \infty\}$. Mostre que C_{GLC} é decidível.

4.28 Seja A uma linguagem Turing-reconhecível consistindo de descrições de máquinas de Turing, $\{\langle M_1 \rangle, \langle M_2 \rangle, \ldots\}$, onde toda M_i é um decisor. Prove que alguma linguagem decidível D não é decidida por nenhum decisor M_i cuja descrição aparece em A. (Dica: Você pode achar útil considerar um enumerador para A.)

SOLUÇÕES SELECIONADAS

4.1 **(a)** Sim. O AFD M aceita 0100.

(b) Não. M não aceita 011.

(c) Não. Essa entrada tem apenas um único componente e, portanto, não é da forma correta.

(d) Não. O primeiro componente não é uma expressão regular e por isso a entrada não é da forma correta.

(e) Não. A linguagem de M não é vazia.

(f) Sim. M aceita a mesma linguagem que si própria.

4.5 **(a)** Não, f não é um-para-um porque $f(1) = f(3)$.

(d) Sim, g é um-para-um.

4.9 A seguinte MT I decide $INFINITA_{\text{AFD}}$.

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

1. Seja k o número de estados de A.
2. Construa um AFD D que aceite todas as cadeias de comprimento k ou mais.
3. Construa um AFD M tal que $L(M) = L(A) \cap L(D)$.
4. Teste $L(M) = \emptyset$, usando o decisor T de V_{AFD} do Teorema 4.4.
5. Se T aceita, *rejeite*; se T rejeita, *aceite*."

Esse algoritmo funciona porque um AFD que aceita uma quantidade infinita de cadeias tem que aceitar cadeias arbitrariamente longas. Por conseguinte, esse algoritmo aceita tais AFDs. Reciprocamente, se o algoritmo aceita um AFD, o AFD aceita alguma cadeia de comprimento k ou mais, onde k é o número de estados do AFD. Essa cadeia pode ser bombeada da maneira prescrita pelo lema do bombeamento para linguagens regulares para se obter uma quantidade infinita de cadeias aceitas.

4.11 A seguinte MT decide A.

"Sobre a entrada $\langle M \rangle$:

1. Construa um AFD O que aceite toda cadeia contendo um número ímpar de 1s.
2. Construa o AFD B tal que $L(B) = L(M) \cap L(O)$.
3. Teste se $L(B) = \emptyset$, usando o decisor T de V_{AFD} do Teorema 4.4.
4. Se T aceita, *aceite*; se T rejeita, *rejeite*."

4.13 Você mostrou no Problema 2.18 que, se C for uma linguagem livre-do-contexto e R uma linguagem regular, então $C \cap R$ é livre-do-contexto. Conseqüentemente, $1^* \cap L(G)$ é livre-do-contexto. A seguinte MT decide A.

"Sobre a entrada $\langle G \rangle$:

1. Construa a GLC H tal que $L(H) = 1^* \cap L(G)$.
2. Teste se $L(H) = \emptyset$, usando o decisor R de V_{GLC} do Teorema 4.8.
3. Se R aceita, *rejeite*; se R rejeita, *aceite*."

4.21 O seguinte procedimento decide $AMBIG_{\text{AFN}}$. Dado um AFN N, construímos um AFD D que simula N e aceita uma cadeia sse ela for aceita por N ao longo de dois ramos de computação diferentes. Aí, então, usamos um decisor para V_{AFD} para determinar se D aceita quaisquer cadeias.

Nossa estratégia para construir D é similar à da conversão de AFN para AFD na prova do Teorema 1.39. Simulamos N mantendo uma pedra sobre cada estado ativo. Começamos colocando uma pedra vermelha sobre o estado inicial e sobre cada estado atingível a partir do inicial ao longo de transições ε. Movemos, adicionamos e removemos pedras de acordo com as transições de N, preservando a cor das pedras. Sempre que duas ou mais pedras são movidas para o mesmo estado, substituímos suas pedras por uma pedra azul. Após ler a entrada, aceitamos se uma pedra azul estiver sobre um estado de aceitação de N.

O AFD D tem um estado correspondente a cada posição possível de pedras. Para cada estado de N, três possibilidades ocorrem: ele pode conter uma pedra vermelha, uma pedra azul ou nenhuma pedra. Por conseguinte, se N tiver n estados,

D terá 3^n estados. Seu estado inicial, seus estados de aceitação e sua função de transição são definidas de modo a realizar a simulação.

4.23 A linguagem de todas as cadeias com igual número de 0s e 1s é uma linguagem livre-do-contexto, gerada pela gramática $S \rightarrow 1S0S \mid 0S1S \mid \varepsilon$. Seja P o AP que reconhece essa linguagem. Construa uma MT M para BAL_{AFD}, que opera da seguinte forma. Sobre a entrada $\langle B \rangle$, onde B é um AFD, use B e P para construir um novo AP R que reconheça a interseção das linguagens de B e P. Aí então, teste se a linguagem de R é vazia. Se sua linguagem for vazia, *rejeite*; caso contrário, *aceite*.