Licenciatura em Engenharia Informática e de Computadores

Teoria da Computação

Exame 1

30 de Junho de 2003

I.1 a) Considere-se a gramática regular $G = (V, I, P, S)$ onde $V = \{S, A, B, C, D\}$, $I = \{x, y\}$ e P contém as produções

$$\begin{array}{lcl} S & \rightarrow & xA \mid yC \\ A & \rightarrow & xA \mid yB \\ B & \rightarrow & xB \mid yA \mid x \\ C & \rightarrow & xD \mid yC \\ D & \rightarrow & yD \mid xC \mid y \end{array}$$

Demonstre-se agora que $xyxxx \in L_G$:

$$\begin{array}{ll} S & \text{símbolo inicial} \\ xA & S \rightarrow xA \\ xyB & A \rightarrow yB \\ xyxB & B \rightarrow xB \\ xyxxB & B \rightarrow xB \\ xyxxx & B \rightarrow x \end{array}$$

I.1 b) A expressão regular obtém-se resolvendo o seguinte sistema de equações:

$$\left\{\begin{array}{l} S = xA + yC \\ A = xA + yB \\ B = xB + yA + x \\ C = xD + yC \\ D = yD + xC + y \end{array}\right. \Leftrightarrow \left\{\begin{array}{l} ----- \\ A = x^*yB \\ ----- \\ C = y^*xD \\ ----- \end{array}\right. \Leftrightarrow \left\{\begin{array}{l} ----- \\ ----- \\ B = xB + yx^*yB + x \\ ----- \\ D = yD + xy^*xD + y \end{array}\right. \Leftrightarrow$$

$$\left\{\begin{array}{l} ----- \\ ----- \\ B = (x + yx^*y)^*x \\ ----- \\ D = (y + xy^*x)^*y \end{array}\right. \Leftrightarrow \left\{\begin{array}{l} ----- \\ A = x^*y(x + yx^*y)^*x \\ ----- \\ C = y^*x(y + xy^*x)^*y \\ ----- \end{array}\right. \Leftrightarrow$$

$$\left\{\begin{array}{l} S = xx^*y(x + yx^*y)^*x + yy^*x(y + xy^*x)^*y \\ ----- \\ ----- \\ ----- \\ ----- \end{array}\right.$$

A expressão regular pretendida é $xx^*y(x + yx^*y)^*x + yy^*x(y + xy^*x)^*y$.

I.1 c) Considere-se a expressão regular $(10^*10^*1 + 0)^*10^*10^*1$, ou a expressão equivalente $(0^*10^*10^*1)^*0^*10^*10^*1$.

I.2 a) Comece-se por verificar se $abc \in L_A$.

$$\delta^*(q_0, abc) = \bigcup_{q' \in \delta^*(q_0, ab)} \delta(q', c) = \delta(q_0, c) \cup \delta(q_2, c) = \emptyset \cup \emptyset = \emptyset$$

$$\delta^*(q_0, ab) = \bigcup_{q' \in \delta^*(q_0, a)} \delta(q', b) = \delta(q_1, b) \cup \delta(q_2, b) \cup \delta(q_3, b) = \{q_0, q_2\} \cup \emptyset \cup \emptyset = \{q_0, q_2\}$$

$$\delta^*(q_0, a) = \bigcup_{q' \in \delta^*(q_0, \epsilon)} \delta(q', a) = \delta(q_0, a) = \{q_1, q_2, q_3\}$$

$$\delta^*(q_0, \epsilon) = \{q_0\}$$

Então $\delta^*(q_0, abc) \cap F = \emptyset \cap \{q_3\} = \emptyset$ pelo que $abc \notin L_A$.

O autómato finito determinista equivalente a A que se obtém por aplicação do algoritmo estudado é $D = (Q', I, \delta', \{q_0\}, F')$ onde $Q' = 2^{\{q_0, q_1, q_2, q_3\}}$, $I = \{a, b, c\}$, $F' = \{X \subseteq \{q_0, q_1, q_2, q_3\} : q_3 \in X\} = \{\{q_3\}, \{q_0, q_3\}, \{q_1, q_3\}, \{q_2, q_3\}, \{q_0, q_1, q_3\}, \{q_0, q_2, q_3\}, \{q_1, q_2, q_3\}, \{q_0, q_1, q_2, q_3\}\}$ e $\delta' : Q' \times I \to Q'$ é definida por

δ'	a	b	c
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$\{q_0\}$	$\{q_1, q_2, q_3\}$	$\emptyset$	$\emptyset$
$\{q_1\}$	$\emptyset$	$\{q_0, q_2\}$	$\{q_3\}$
$\{q_2\}$	$\{q_1, q_2, q_3\}$	$\emptyset$	$\emptyset$
$\{q_3\}$	$\emptyset$	$\emptyset$	$\emptyset$
$\{q_0, q_1\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2\}$	$\{q_3\}$
$\{q_0, q_2\}$	$\{q_1, q_2, q_3\}$	$\emptyset$	$\emptyset$
$\{q_0, q_3\}$	$\{q_1, q_2, q_3\}$	$\emptyset$	$\emptyset$
$\{q_1, q_2\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2\}$	$\{q_3\}$
$\{q_1, q_3\}$	$\emptyset$	$\{q_0, q_2\}$	$\{q_3\}$
$\{q_2, q_3\}$	$\{q_1, q_2, q_3\}$	$\emptyset$	$\emptyset$
$\{q_0, q_1, q_2\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2\}$	$\{q_3\}$
$\{q_0, q_1, q_3\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2\}$	$\{q_3\}$
$\{q_0, q_2, q_3\}$	$\{q_1, q_2, q_3\}$	$\emptyset$	$\emptyset$
$\{q_1, q_2, q_3\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2\}$	$\{q_3\}$
$\{q_0, q_1, q_2, q_3\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2\}$	$\{q_3\}$

I.2 b) Obviamente o autómato D tem estados inúteis, nomeadamente, o estado $\emptyset$. Logo é claro que existe um autómato finito D' com menos estados do que D tal que $L_{D'} = L_D$. Para obter D', começa-se por eliminar de D todos os estados inúteis. Por análise da tabela acima e tendo em conta que $\{q_0\}$ é o estado

inicial, facilmente se conclui que só são relevantes os estados $\{q_0\}$, $\{q_1, q_2, q_3\}$, $\{q_0, q_2\}$ e $\{q_3\}$, e que todos estes são produtivos.

Aplica-se agora o algoritmo de procura exaustiva de pares de estados distinguíveis.

$\{q_1, q_2, q_3\}$	$\times_F$	—	—
$\{q_0, q_2\}$		$\times_F$	—
$\{q_3\}$	$\times_F$	$\times_{abc}$	$\times_F$
	$\{q_0\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2\}$

Começa-se por marcar como distinguíveis usando $\times_F$ os pares em que um dos estados é final, $\{q_1, q_2, q_3\}$ ou $\{q_3\}$, e o outro não, $\{q_0\}$ ou $\{q_0, q_2\}$.

De seguida marca-se também o par formado por $\{q_1, q_2, q_3\}$ e $\{q_3\}$ usando $\times_{abc}$ pois o primeiro tem transições definidas para a, b e c e o segundo não.

No passo iterativo nada mais é marcado. Em particular, o par formado por $\{q_0\}$ e $\{q_0, q_2\}$ não é marcado pois ambos têm apenas uma transição pelo símbolo a e ambas levam ao mesmo estado $\{q_1, q_2, q_3\}$.

Conclui-se que os estados $\{q_0\}$ e $\{q_0, q_2\}$ são equivalentes. Logo, tomando $S = \{\{q_0\}, \{q_0, q_2\}\}$, $T = \{\{q_1, q_2, q_3\}\}$ e $R = \{\{q_3\}\}$, pode construir-se o autómato mínimo $D = (\{S, T, R\}, \{a, b, c\}, \delta'', S, \{T, R\})$ onde $\delta'' : \{S, T, R\} \times \{a, b, c\} \to \{S, T, R\}$ não total é definida por

δ''	a	b	c
S	T		
T	T	S	R
R			

I.3 a) Considere-se o programa URM seguinte:

1. DEZ[5]
2. S[6]
3. S[2]
4. DIGN[2, 1, 3]
5. J[3, 5, 11]
6. PROD[3, 6, 3]
7. SUM[3, 4, 4]
8. PROD[5, 6, 6]
9. S[2]
10. J[1, 1, 3]
11. T[4, 1]

I.3 b) Pretende-se calcular o programa URM cujo número de Gödel é 2561, ou seja, $\gamma^{-1}(2561)$. Toma-se 2561+1=2562 e converte-se em binário 101000000010. Da direita para a esquerda tem-se 1 zero até ao primeiro um, depois 7 zeros até ao segundo um, e finalmente 1 zero até ao último um. Logo o programa pretendido é $\langle C_1, C_2, C_3\rangle$ onde $C_1 = C_3 = \beta^{-1}(1)$ e $C_2 = \beta^{-1}(7)$.

Para calcular $\beta^{-1}(1)$, note-se que $1 = 4 \times 0 + 1$ pelo que $C_1 = C_3 = \mathrm{S}[n]$ com $n - 1 = 0$, ou seja, $n = 1$.

Para calcular $\beta^{-1}(7)$, note-se que $7 = 4 \times 1 + 3$ pelo que $C_2 = \mathrm{J}[m, n, q]$ com $\xi(m, n, q) = \pi(\pi(m - 1, n - 1), q - 1) = 1$. Como $1 + 1 = 2 = 2^1(2 \times 0 + 1)$, tem-se que $\pi(m - 1, n - 1) = 1$ e $q - 1 = 0$ pelo que $q = 1$. Para calcular m e n note-se de novo que $1 + 1 = 2 = 2^1(2 \times 0 + 1)$ e portanto tem-se $m - 1 = 1$ e $n - 1 = 0$, ou seja, $m = 2$ e $n = 1$.

O programa pretendido é portanto $\langle \mathrm{S}[1], \mathrm{J}[2, 1, 1], \mathrm{S}[1]\rangle$.

A função binária calculada pelo programa é caracterizada por

$$\lambda xy. \begin{cases} x + 3 & \text{se } y = x + 1 \\ x + 2 & \text{se } y \neq x + 1 \end{cases}$$

II.1 a) Seja L a linguagem das sequências do alfabeto $\{0, 1\}$ cujo número de 0's é o dobro do número de 1's, e suponha-se por absurdo que L é regular. Então existe um autómato finito determinista $D = (Q, \{0, 1\}, \delta, q_o, F)$ tal que $L_D = L$. Considere-se então $n = \#Q$ e tome-se a sequência $w = 0^{2n}1^n \in L_D$.

Como $|w| = 2n + n = 3n \geq n$ o lema da bombagem garante que existem sequências w_1, w_2 e w_3 tais que $w = w_1w_2w_3$, $w_2 \neq \epsilon$, $|w_1w_2| \leq n$ e $w_1w_2^rw_3 \in L_D$ para qualquer $r \geq 0$.

Sendo $w = w_1w_2w_3$ e $|w_1w_2| \leq n$ facilmente se conclui que w_1 e w_2 só têm 0's. Ter-se-á portanto $w_1 = 0^i$, $w_2 = 0^j$ e $w_3 = 0^k1^n$ com $i + j \leq n$, $j \neq 0$ e $i + j + k = 2n$.

Tome-se por exemplo $r = 0$. Então $w_1w_2^0w_3 = w_1w_3 = 0^i0^k1^n = 0^{i+k}1^n \in L_D = L$, podendo-se concluir que $i + k = 2n$. Mas como também $i + j + k = 2n$ deverá ter-se $j = 0$ o que contradiz $j \neq 0$.

Conclui-se que D não existe e portanto a linguagem L não é regular.

II.1 b) Dada a gramática regular $G = (V, I, P, S)$ pode construir-se um autómato finito não determinista A tal que $L_A = L_G$ conforme um dos dois casos descritos de seguida.

Se P não contém produções da forma $X \to i$, constrói-se $A = (V, I, \delta, S, F)$ com $F = \{R \in V : R \to \epsilon \in P\}$ e define-se

$$\delta(R, j) = \{T \in V : R \to jT \in P\}$$

Se P contém produções da forma $X \to i$, constrói-se $A = (V \cup \{X_F\}, I, \delta, S, F)$ com $F = \{R \in V : R \to \epsilon \in P\} \cup \{X_F\}$ e define-se

$$\delta(R, j) = \begin{cases} \{T \in V : R \to jT \in P\} & \text{se } R \to j \notin P \\ \{T \in V : R \to jT \in P\} \cup \{X_F\} & \text{se } R \to j \in P \end{cases}$$

II.1 c) Em geral, para obter a partir de $A = (Q, I, \delta, q_0, F)$ um autómato finito não determinista $\overline{A}$ tal que $L_{\overline{A}} = I^* \setminus L_A$, pode proceder-se da seguinte forma:

1. Começa-se por obter um autómato finito determinista D equivalente a A, tomando $D = (2^Q, I, \delta', \{q_0\}, \{X \subseteq Q : X \cap F \neq \emptyset\})$ onde se define $\delta'(X, i) = \bigcup_{q \in X} \delta(q, i)$. Obviamente, $L_D = L_A$.
2. Notando que a função δ' é total, constrói-se o complementar de D, ou seja o autómato finito determinista $\overline{D} = (2^Q, I, \delta', \{q_0\}, \{X \subseteq Q : X \cap F = \emptyset\})$. Obviamente $L_{\overline{D}} = I^* \setminus L_D$.
3. Sendo $\overline{D}$ determinista, basta agora formulá-lo como não determinista, obtendo $\overline{A} = (2^Q, I, \overline{\delta}, \{q_0\}, \{X \subseteq Q : X \cap F = \emptyset\})$ com $\overline{\delta}(X, i) = \{\delta'(X, i)\}$. Claramente, $L_{\overline{A}} = L_{\overline{D}} = I^* \setminus L_D = I^* \setminus L_A$.

II.2 a) A função definida por minimização a partir de $\lambda xy.\,|2y - x|$ é

$$\lambda x.\mu y.(|2y - x| = 0)$$

Fixado x, $\mu y.(|2y - x| = 0)$ define, se existir, o menor y tal que $|2y - x| = 0$, ou seja, $2y = x$, e $|2z - x|$ definido para $0 \leq z < y$. Esta última condição não levanta qualquer problema pois $\lambda xy.\,|2y - x|$ é uma função total.

Então, obviamente, existe y tal que $2y = x$ se e só se x é par, caso em que necessariamente $y = x/2$. Logo, a função definida por minimização é precisamente

$$\lambda x. \begin{cases} x/2 & \text{se } x \text{ é par} \\ \text{indef} & \text{se } x \text{ é ímpar} \end{cases}$$

II.2 b) Dada a enumeração efectiva $\gamma : \mathbb{P} \to I\!N_o$ dos programas URM, e denotando por ϕ_n a função de 1 argumento calculada pelo programa $\gamma^{-1}(n)$, tem-se que a sequência infinita $\phi_0, \phi_1, \phi_2, \ldots$ é também uma enumeração das funções unárias computáveis. Usando o argumento diagonal de Cantor constrói-se a função $f : I\!N_o \to I\!N_o$ da seguinte forma:

$$f(k) = \begin{cases} \phi_k(k) + 1 & \text{se } \phi_k(k) \downarrow \\ 0 & \text{se } \phi_k(k) \uparrow \end{cases}$$

Claramente, tem-se que $f \neq \phi_0, f \neq \phi_1, f \neq \phi_2, \ldots$. Nomeadamente tem-se que $f \neq \phi_k$ pois $f(k) \neq \phi_k(k)$. Em particular, se $\phi_k(k) \downarrow$ então $f(k) = \phi_k(k) + 1 \neq \phi_k(k)$, e se $\phi_k(k) \uparrow$ então $f(k) = 0 \neq \phi_k(k)$.

Conclui-se que f não é computável.