Teoria da Computação
Resolução do Exame 1 (LESIM e LERCI)
2002/2003

Grupo 1:

1.1

a) $D = (Q, I, \delta, q_0, F)$

- $Q = \{q_0, q_1, q_2, q_3\}$
- $I = \{a, b, c\}$
- $F = \{q_2, q_3\}$
- $\delta : Q \times I \to Q$ tal que

δ	a	b	c
q_0	q_1	q_2	q_0
q_1	nd	q_2	q_1
q_2	q_2	nd	q_3
q_2	q_2	nd	nd

b) O autómato D não tem estado inúteis. Assim, para saber se existe um autómato finito determinista com menos estados que D que reconheça exactamente L_D, há apenas que determinar se existem pares de estados equivalentes em D. Para tal utiliza-se o algoritmo de procura exaustiva de pares distinguíveis. A função δ não é total mas todos os estados são, em particular, produtivos pelo que o algoritmo pode ser utilizado. Constrói-se então a tabela seguinte (onde $\surd$ e $\surd_1$ identificam pares de estados distinguíveis).

q_1	$\surd$		
q_2	$\surd_1$	$\surd_1$	
q_3	$\surd_1$	$\surd_1$	$\surd$
	q_0	q_1	q_2

Os pares de estados identificados com $\surd_1$ são pares de estados distinguíveis porque um dos estados é final e o outro não é.

Dado que $\delta(q_0, a)$ está definido e $\delta(q_1, a)$ não está, $\{q_0, q_1\}$ também é um par de estados distinguíveis. Como $\delta(q_2, c)$ está definido e $\delta(q_3, c)$ não está, $\{q_2, q_3\}$ também é um par de estados distinguíveis.

Como todos os pares de estados foram já identificados como distinguíveis, não é necessário aplicar o passo iterativo.

Dado que não há pares de estados equivalentes, não existe nenhum autómato finito determinista com menos estados que D que reconheça exactamente L_D.

c) Tendo em conta o algoritmo estudado tem-se que $\overline{D} = (Q', I, \delta', q_0, F')$

- $Q' = \{q_0, q_1, q_2, q_3, q\}$

- $I = \{a, b, c\}$
- $F' = \{q_0, q_1, q\}$
- $\delta' : Q' \times I \to Q'$ tal que

δ	a	b	c
q_0	q_1	q_2	q_0
q_1	q	q_2	q_1
q_2	q_2	q	q_3
q_2	q_2	q	q
q	q	q	q

1.2
a) $G = (V, I, P, S)$

- $V = \{S, A, B, C, D\}$
- $I = \{a, b, c\}$
- $P ::\ \begin{array}{l} S \to aA \mid bB \\ A \to aA \mid bA \mid cC \\ B \to aB \mid bB \mid cD \\ C \to aC \mid bC \mid cC \mid a \\ D \to aD \mid bD \mid cD \mid b \end{array}$

b)

1. S símbolo inicial
2. aA $S \to aA$
3. abA $A \to bA$
4. $abcC$ $A \to cC$
5. $abca$ $C \to a$

c)

$$\begin{cases} S = aA + bB \\ A = aA + bA + cC \\ B = aB + bB + cD \\ C = aC + bC + cC + a \\ D = aD + bD + cD + b \end{cases}$$

$$\begin{cases} S = aA + bB \\ A = (a+b)A + cC \\ B = (a+b)B + cD \\ C = (a+b+c)C + a \\ D = (a+b+c)D + b \end{cases}$$

$$\begin{cases} S = aA + bB \\ A = (a+b)^*cC \\ B = (a+b)^*cD \\ C = (a+b+c)^*a \\ D = (a+b+c)^*b \end{cases}$$

$$\begin{cases} S = aA + bB \\ A = (a+b)^*c(a+b+c)^*a \\ B = (a+b)^*c(a+b+c)^*b \\ C = (a+b+c)^*a \\ D = (a+b+c)^*b \end{cases}$$

$$\begin{cases} S = a(a+b)^*c(a+b+c)^*a + b(a+b)^*c(a+b+c)^*b \\ A = (a+b)^*c(a+b+c)^*a \\ B = (a+b)^*c(a+b+c)^*b \\ C = (a+b+c)^*a \\ D = (a+b+c)^*b \end{cases}$$

A expressão regular pedida é $a(a+b)^*c(a+b+c)^*a+b(a+b)^*c(a+b+c)^*b$.

1.3 A expressão regular pedida é $b(a+b+c)^*aa(a+b+c)^*+c(b+c+a(b+c)^*a)^*$.

1.4
a) A função ternária calculada pelo programa é a função $\lambda xyz.max(x+y,z)$. Havia que fazer o fluxograma correspondente ao programa.

b) Uma possível solução é:
1. $J(1,3,5)$
2. $J(2,3,5)$
3. $S(3)$
4. $J(1,1,1)$
5. $S(5)$
6. $J(3,4,10)$
7. $S(4)$
8. $DOBRO[5,5]$
9. $J(1,1,6)$
10. $T(5,1)$

c) Pretende-se calcular $\gamma^{-1}(20511)$ (ou seja o programa $P =< p_1,\ldots,p_n >$ tal que $\gamma(P) = \tau(\beta(p_1),\ldots,\beta(p_n)) = 20511$):

- Cálculo de $\tau^{-1}(20511)$: $20511 \stackrel{+1}{\to} 20512 = 2^5 + 2^{12} + 2^{14}$ o que significa que

- P tem três comandos
- $b_1 = 5$, $b_2 = 12$ e $b_3 = 14$ logo
    - $a_1 = b_1 = 5$
    - $a_2 = b_2 - b_1 - 1 = 6$
    - $a_3 = b_3 - b_2 - 1 = 1$

e portanto $\tau^{-1}(20511) = (5, 6, 1)$ o que significa que
$\gamma^{-1}(20511) = (\beta^{-1}(5), \beta^{-1}(6), \beta^{-1}(1))$

- Cálculo de $\beta^{-1}(1)$: $1 = 4 \times u + v = 4 \times 0 + 1$ logo
    - como $v = 1$ o comando é de tipo S
    - como $u = 0$ o comando é $S(1)$ (ou seja, $S(u + 1)$)
- Cálculo de $\beta^{-1}(5)$: $1 = 4 \times u + v = 4 \times 1 + 1$ logo
    - como $v = 1$ o comando é de tipo S
    - como $u = 1$ o comando é $S(2)$ (ou seja, $S(u + 1)$)
- Cálculo de $\beta^{-1}(6)$: $6 = 4 \times u + v = 4 \times 1 + 2$ logo
    - como $v = 2$ o comando é de tipo T
    - como $u = 1$ o comando é $T(m, n)$ onde $m = \pi_1(1)+1$ e $n = \pi_2(1)+1$ que se calculam da seguinte forma
        - cálculo de $\pi_1(1)$ e $\pi_2(1)$: $1 \overset{+1}{\to} 2 = 2^1 \times 1$

$$1 \longleftarrow 2^1 \times 1 \longrightarrow (1 - 1)/2 = 0$$

e portanto
$\pi_1(1) = 1$
$\pi_2(1) = 0$
o que significa que $m = 2$ e $n = 1$

Conclui-se assim que $\gamma^{-1}(20511) =< S(2), T(2, 1), S(1) >$.

Grupo 2:
2.1 O autómato é $A' = (Q_1 \cup Q_2 \cup \{q\}, I, \delta', q, F_1 \cup F_2)$, com $q \notin Q_1 \cup Q_2$, no qual $\delta' : Q \times (I \cup \epsilon) \to 2^Q$ é tal que, para cada $q' \in Q_1 \cup Q_2 \cup \{q\}$ e $i \in I$,

- $\delta'(q, \epsilon) = \{q_0^1, q_0^2\}$
- $\delta'(q, i) = \emptyset$
- $\delta'(q', i) = \delta_1(q', i)$ se $q' \in Q_1$
- $\delta'(q', i) = \delta_2(q', i)$ se $q' \in Q_2$.

2.2 A função h é definida por recursão a partir da função $f : I\!N_0 \to I\!N_0$ tal que $f(x) = 1$ e da função $g : I\!N_0^3 \to I\!N_0$ tal que $g(x, y, z) = z \times x$ porque

- $h(x,0) = x^0 = 1 = f(x)$ e
- $h(x,y+1) = x^{y+1} = x \times x^y = g(x,y,h(x,y))$.

2.3 O problema da paragem pode ser enunciado da seguinte forma: " dados $m,n \in \mathbb{N}_0$, determinar se $\phi_n(m)$ está definido" (uma outra forma de enunciar o problema é: "dados $m,n \in \mathbb{N}_0$, determinar se $m \in W_n$").

Prova-se agora que o problema da paragem não é decidível. A prova decorre por absurdo. Suponha-se que o predicado "$\phi_n(m)$ está definido" é decidível. Então a sua função característica $c : \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$ tal que

$$c(n,m) = \begin{cases} 1 & \text{se } \phi_n(m)\downarrow \\ 0 & \text{caso contrário} \end{cases}$$

é URM-computável. Considerando a função $f : \mathbb{N}_0 \to \mathbb{N}_0$ tal que

$$f(n) = \begin{cases} 0 & \text{se } c(n,n) = 0 \\ \text{não definida} & \text{caso contrário} \end{cases}$$

tem-se que se c é computável então f também é. Então $f = \phi_k$ para algum $k \in \mathbb{N}_0$. Tem-se que $c(k,k) = 1$ sse $\phi_k(k)\downarrow$ sse $f(k)\uparrow$ sse $\phi_k(k)\uparrow$ sse $c(k,k) = 0$, ou seja, chega-se a um absurdo. Conclui-se assim que c não pode ser computável. e portanto o problema da paragem não é decidível.