Prática 7 - Properties of regular languages

1) a) Vamos assumir que L é uma linguagem regular. Se é uma linguagem regular, tem de satisfazer o lema da bombagem para linguagens regulares.
Escolhemos $w = 0^m 1^{2m}$, cujo comprimento, $|w| = 3m$, é $\geq m$.
Então, $w = xyz$, e como $|xy| \leq m$ e $|y| \geq 1$, então y só pode ter 0s, e $1 \leq |y| \leq m$. Logo, xyz, pode ser representado por:
$0^j 0^t 0^{m-t-j} 1^{2m}$, com $1 \leq t \leq m$ e $0 \leq j \leq m-1$.
Como para todo o $K \geq 0$, a cadeia $xy^k z$ também está em L, se fizermos $K = 0$ temos:
$0^j 0^{m-t-j} 1^{2m}$, ou seja $2m$ 1s e $m-t$ 0s. E como $1 \leq t \leq m$, o número de 1s será sempre maior do que o dobro dos 0s e por isso essas cadeias (dadas por $0^j 0^{m-t-j} 1^{2m}$) não pertencem a L.
A linguagem não pode então ser regular.

b) $L = (0+1)^m 1^m$ Vamos assumir que L é uma linguagem regular. Se é uma linguagem regular, tem de satisfazer o lema da bombagem para linguagens regulares.
Escolhemos $w = 0^m 1^m$, cujo comprimento, $|w| = 2m$, é $\geq m$.
Então, $w = xyz$, e como $|xy| \leq m$ e $|y| \geq 1$, então y só pode ter zeros, e $1 \leq |y| \leq m$. Logo, xyz, pode ser representado por:
$0^i 0^j 0^{m-i-j} 1^m$, com $1 \leq j \leq m$ e $0 \leq i \leq m-1$.
Como para todo o $k \geq 0$, a cadeia $xy^k z$ também está em L, se fizermos $k = 0$, temos:
$0^i 0^{m-i-j} 1^m$, ou seja, m 1s e $m-j$ 0s. E como $1 \leq j \leq m$, o número de 1s será sempre maior do que o número de 0s, e por isso essas cadeias não pertencem a L, pelo que L não é regular.

e) Vamos assumir que L é uma linguagem regular.
Escolhemos $w = (00+11)^m$, cujo comprimento, $|w| = 2m$, é $\geq m$.
Então, $w = xyz$, e como $|xy| \leq m$ e $|y| \geq 1$, então pode ser representado por:

d) Vamos assumir que L é uma linguagem regular. Escolhemos $w = 01^m0^m1$, cujo comprimento é $\geq m$. Então, $w = xyz$, e como $|xy| \leq m$ e $|y| \geq 1$, ...

e) Vamos assumir que L é uma linguagem regular. Se é uma linguagem regular, tem que satisfazer o lema da bombagem para linguagens regulares. Escolhemos $w = 0^{m^2}$. $w = xyz$ e $|w| = m^2$, logo $|w| \geq m$. Como $|xy| \leq m$, y contem entre 1 a m 0s ($1 \leq |y| \leq m$). Para $k = 2$, temos:

$m^2 + 1 \leq |xyyz| \leq m^2 + m$. O quadrado perfeito a seguir a m^2 é $(m+1)^2 = m^2 + 2m + 1 > m^2 + m$.

Logo, $|xyyz|$ não pode ser um quadrado perfeito! Mas $xyyz$ deveria fazer parte da linguagem L e não faz! Logo, L não é uma linguagem regular.

2 a) Se L é uma linguagem regular, então existe um DFA que representa L. Pode-se construir o DFA de L/a a partir do DFA de L em que definimos todos os estados como de não aceitação e em seguida para cada estado q do DFA marcamos q como estado de aceitação se $\delta(q, a)$ for um estado de aceitação no DFA de L. Logo, se é possível obter um DFA para L/a, então é ling. regular.

b) Aplicando as operações de reverso e quociente conseguimos obter $a \backslash L$ a partir de L e por isso $a \backslash L$ tem de ser uma linguagem regular.

$$L \rightarrow L_{rev} \rightarrow L_{rev}/a \rightarrow (L_{rev}/a)_{rev} \rightarrow a \backslash (L_{rev})_{rev} = a \backslash L$$

e) • Falso

• Falso

• Verdadeiro

• Verdadeiro.

③ Seja $L_1 = \{0^i 1^j, i \neq j\}$

Podemos obter $L_{0^n1^n}$ usando as seguintes operações fechadas sobre as linguagens regulares:

$$L_{0^n1^n} = \overline{L_1} \cap L(0^*1^*)$$

Como $L(0^*1^*)$ é uma linguagem regular, se L_1 fosse regular então o complemento de L_1 também seria uma linguagem regular, e como $L(0^*1^*)$ é uma linguagem regular, a intersecção das duas linguagens $(\overline{L_1} \cap L(0^*1^*))$ também originaria uma linguagem regular.

Como sabemos que $L_{0^n1^n}$ não é uma linguagem regular então L_1 não pode ser uma linguagem regular.

④ a) Testar todas as strings de comprimento entre n e $2n-1$ (n é o número de estados do DFA). Se alguma string pertencer a L então L é uma linguagem infinita.

Se não existir nenhuma string de comprimento entre n e $2n-1$ aceite também não há strings aceites de tamanho $\geq 2n$ e portanto a linguagem L é finita (todas as strings aceites são de tamanho $< n$).

Outro Método: Identificar ciclos no grafo do DFA, mas teria de se ter cuidado de forma a que os ciclos identificados contribuíssem para a aceitação de strings, ou seja, que estivessem em caminho do estado inicial para um dos estados de aceitação.

b) Pode fazer-se o complemento do DFA da linguagem e verificar que não aceita nenhuma string.

Testar o DFA complemento com todas as strings de tamanho entre n e $2n-1$ (n é um número de estados do DFA). Se nenhuma for aceite então o complemento de L é uma linguagem vazia e L é a linguagem que aceita todas as strings no alfabeto.

Verificar se o complemento de um DFA não aceita strings pode ser feito analisando se existem caminhos no DFA do estado de entrada para os estados de aceitação.

5) a)

B	X							
C	X	X						
D		X	X					
E	X		X	X				
F	X	X		X	X			
G		X	X		X	X		
H	X		X	X		X	X	
I	X	X		X	X		X	X
	A	B	C	D	E	F	G	H

b)

	0	1
→{A,D,G}	{B,E,H}	{B,E,H}
{B,E,H}	{C,F,I}	{C,F,I}
*{C,F,I}	{A,D,G}	{B,E,H}

6) a)

B				
C	X	X		
D	X	X	X	
E	X	X		X
	A	B	C	D

DFA1 minimizado:

	a	b
→*{A,B}	{A,B}	{C,E}
*{C,E}	{D}	{C,E}
{D}	{D}	{D}

b) Se renomearmos os estados do DFA1 minimizado para {A,B} → F, {C,E} → G e {D} → H obtemos o DFA:

	a	b
→*F	F	G
*G	H	G
H	H	H

Que é precisamente igual à tabela de transições do DFA2. Os DFAs são por isso equivalentes.

(Outro método: (construir a tabela de estados distinguíveis para os 2).

*B							
*C	X	X					
D	X	X	X				
*E	X	X		X			
→*F			X	X	X		
*G	X	X		X		X	
H	X	X	X		X	X	X
	→*A	*B	*C	D	*E	*F	*G

Logo, os estados de início F e A não são distinguíveis (são equivalentes). E isso indica que os DFAs são equivalentes (condição necessária e suficiente).