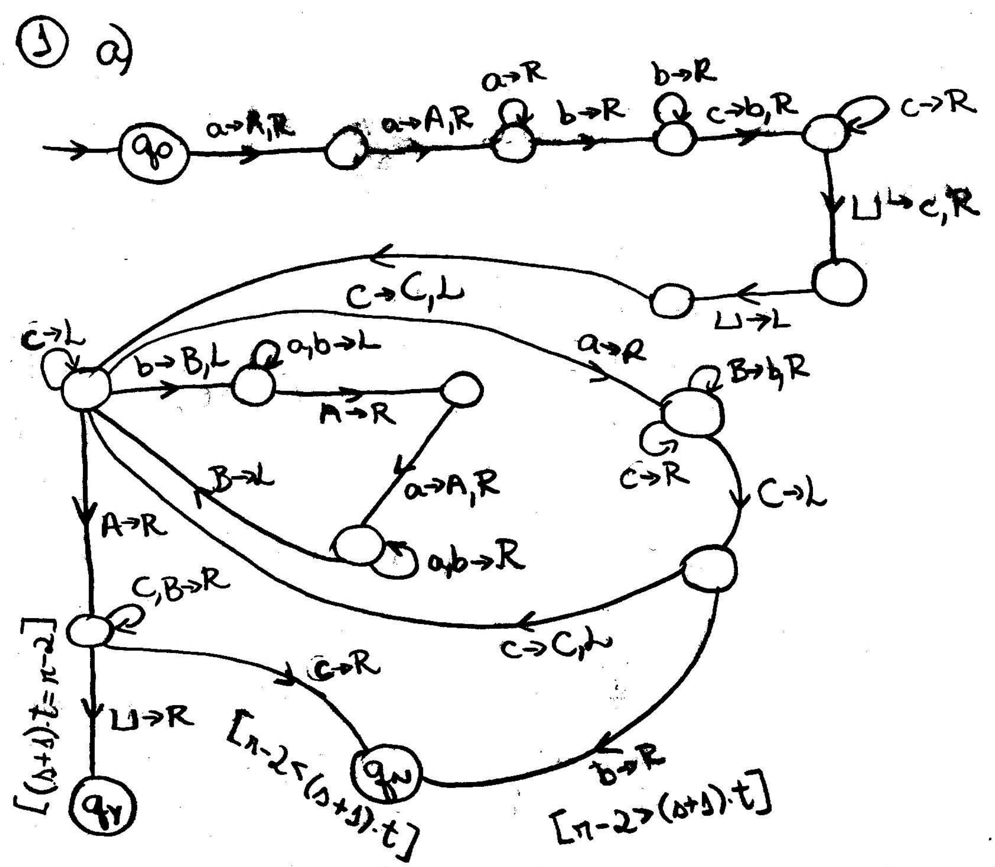
UFLA – Universidade Federal de Lavras
Departamento de Ciência da Computação
COM167 – Teoria da Computação
Prof. Rudini Sampaio
Monitor: Rodrigo Pereira dos Santos

Primeira Lista de Exercícios – **2005/1**

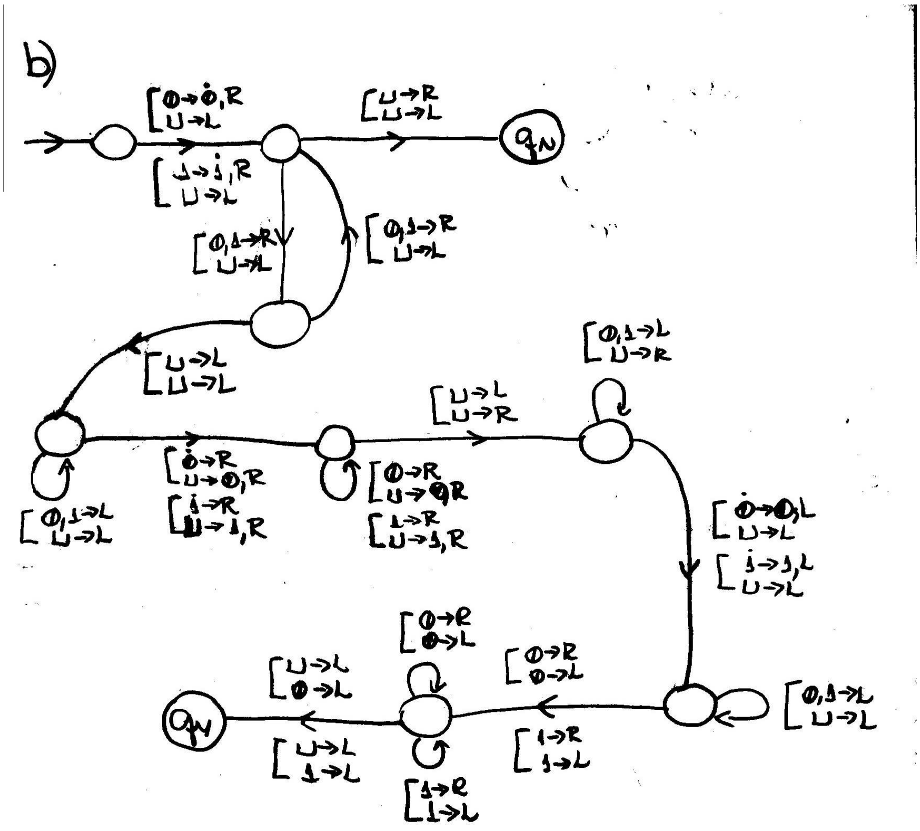
Exercício 1 Desenhe Diagrama de Estados para Máquinas que Decidem as Linguagens:

a. M.T. det. com uma fita para $L_1 = \{a^r b^s c^t \mid r, s, t > 0 \text{ e } (s+1) \bullet t = r-2\}$



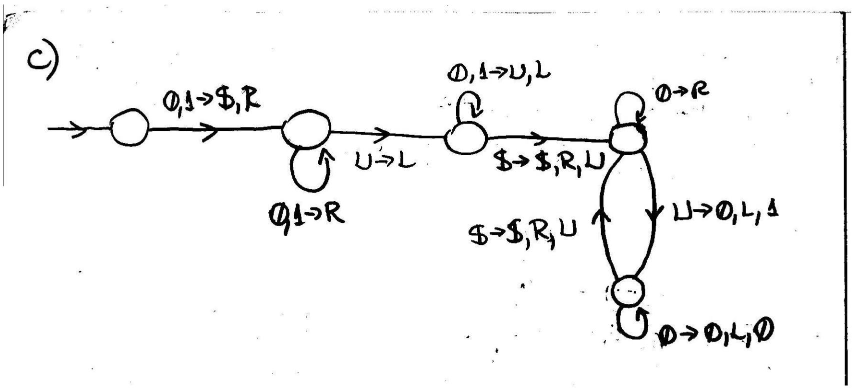
As transições não mencionadas levam ao estado de rejeição q_N.

b. M.T. det. com duas fitas para $L_2 = \{ww^R \mid w \in (0,1)^*$, onde w^R é o reverso de $w\}$



As transições não mencionadas levam ao estado de rejeição q_N.

c. Enumerador para $L_3 = \{10^*\}$, ou seja, todas as palavras iniciando em 1 e seguido de zero ou mais 0s.



....

Exercício 2 Obtenha a função *f* de correspondência entre os Naturais e os Racionais, conforme visto em sala. Obtenha a inversa de *f*. Justifique.

RESPOSTA =

Obs.) Vamos mostrar a função de correspondência entre IN e QP (racionais não negativos). Para QN (racionais negativos), a demonstração é análoga.

O conjunto dos números racionais não negativos QP é enumerável, como mostra a função sobrejetora g: IN → QP, que será definida de forma a espelhar a correspondência mostrada esquematicamente na figura abaixo:

	1	2	3	4	5	⋯
0	0	1	3	6	10	
1	2	4	7	11		
2	5	8	12			
3	9	13				
4	14					
⋮						

Figura 1.5: Função sobrejetora de $\mathbf{N}$ para QP.

Nesta figura, os números naturais são dispostos em uma matriz infinita com linhas numeradas 0, 1, 2, ..., e as colunas 1, 2, 3, ... Observe que, começando na posição (0,1) da matriz, os naturais são dispostos nas diagonais inversas, em seqüência, a partir de 0. Com este esquema, todo natural terá uma posição na matriz. Supondo que o par (linha i, coluna j) represente o número racional i/j, observe que cada racional possui múltiplas representações. Por exemplo, o número 0 tem inúmeras representações: 0/1, 0/2, 0/3, etc. O número 1: 1/1, 2/2, etc. A matriz representa uma função g tal que g(k) = i/j, onde k é o natural especificado na linha i, coluna j. Isto é suficiente para concluir que o conjunto QP é enumerável.

Antes de determinar g, será determinada a função f: IN x IN – {0} → IN, onde f(i, j) = k se g(k) = i/j. Observe, inicialmente, que f(i, j) = S + i, onde S é o primeiro natural da diagonal inversa em que se situa k (na linha 0). Para os elementos de uma diagonal inversa, i + j é constante e é exatamente a quantidade de números naturais dispostos na diagonal. Assim, S é a quantidade de números naturais já colocados antes da diagonal, que é:

$$\sum_{k=0}^{i+j-1} k = \frac{(i+j)^2}{2} - \frac{i}{2} - \frac{j}{2} = (i + j)(i + j - 1)/2$$

Logo, f(i, j) = (i + j)(i + j – 1)/2 + i.

A inversa desta função dá o par (i, j) para o qual g(k) = i/j, para cada natural k. Dado um número k, deve-se, assim, determinar i e j tais que k = (i + j)(i + j – 1)/2 + i, ou seja, 2k – 2i = (i + j)(i + j - 1). Veja que i +j deve ser o maior número natural tal que (i + j)(i + j – 1) ≤ 2k. Assim, designando-se por <u>n</u> o maior número natural tal que n(n – 1) ≤ 2k, determina-se i através de i = k – n(n – 1)/2 e, em seguida, determina-se j através de j = n – i.

■

....

Exercício 3 [3.6 Sipser] O Teorema 3.13 diz que uma linguagem é recursivamente enumerável se e somente se algum Enumerador a enumera. Porque não usamos na prova o seguinte algoritmo mais simples? (Como na prova do teorema, s_1, s_2, s_3,..., é uma lista das palavras de Σ^* em ordem lexicográfica).

E = "Ignore a Entrada

1. Repita para i=1, 2, 3, ...

1.1. Rode M com entrada s_i

1.2. Se M *aceita*, *imprima* s_i"

RESPOSTA = O problema está no passo (1.1), ao simular M sem colocar um limite no número de passos. Não foi dito que a M.T. M decide a linguagem, então a M.T. pode entrar em *loop*, já que a linguagem é recursivamente enumerável.

....

Exercício 4 [3.14 Sipser] Mostre que a classe de linguagens decididas por Máquinas de Turing, ou seja, a classe de linguagens recursivas, é fechada sob as operações de:

RESPOSTA =

Sejam L_1 e L_2 linguagens recursivas, tais que existem M.T.'s M_1 e M_2 que decidem L_1 e L_2, respectivamente.

a. União

Vamos construir uma M.T. M_3 que decide a linguagem $L_3 = L_1 \cup L_2$.

M_3 = "Com entrada w:

1. Simule M_1 com entrada w.
2. Simule M_2 com entrada w.
3. Se M_1 ou M_2 aceitam, ACEITE.
   Senão, REJEITE."

■

b. Intersecção

Vamos construir uma M.T. M_4 que decide a linguagem $L_4 = L_1 \cap L_2$.

M_4 = "Com entrada w:

1. Simule M_1 com entrada w.
2. Simule M_2 com entrada w.
3. Se M_1 e M_2 aceitam, ACEITE.
   Senão, REJEITE."

■

c. Concatenação

Vamos construir uma M.T. M_5 que decide a linguagem $L_5 = L_1 \bullet L_2$.

M_5 = “Com entrada $w = w_1w_2w_3...w_n$, onde $n = |w|$ e w_i é cada um dos caracteres de w:

1. Para i de 0 até n, faça:

1.1 Simule M_1 com entrada $w_1w_2...w_i$.

1.2 Simule M_2 com entrada $w_{i+1}...w_n$.

1.3 Se M_1 e M_2 aceitam, ACEITE.
Senão, REJEITE.”

■

d. Estrela

Vamos construir uma M.T. M_6 que decide a linguagem $L_6 = L_1^*$.

M_6 = “Com entrada $w = w_1w_2w_3...w_n$, onde $n = |w|$ e w_i é cada um dos caracteres de w:

1. Se $w = \varepsilon$, ACEITE.
2. Senão, para i de 0 até n faça:

2.1 Simule M_1 com entrada $w_1w_2...w_i$.

2.2 Se M_1 aceita, então:

2.2.1 Simule M_6 com entrada $w_{i+1}...w_n$.

2.2.2 Se M_6 aceita, ACEITE.

3. REJEITE.”

■

e. Complementação

Vamos construir uma M.T. M_7 que decide a linguagem $L_7 = \neg L_1$ (complemento ou negação de L_1).

M_7 = “Com entrada w:

1. Simule M_1 com entrada w.
2. Se M_1 aceita, REJEITE.
Senão, ACEITE.”

■

....

Exercício 5 [3.15 Sipser] Mostre que a classe de linguagens reconhecidas por Máquinas de Turing, ou seja, a classe de linguagens recursivamente enumeráveis, é fechada sob as operações de:

RESPOSTA =

Sejam L_1 e L_2 linguagens recursivamente enumeráveis, tais que existem M.T.’s M_1 e M_2 que reconhecem L_1 e L_2, respectivamente.

a. União

Vamos construir uma M.T. M_3 que reconhece a linguagem $L_3 = L_1 \cup L_2$.

M_3 = "Com entrada w:
1. Para i = 1, 2, 3, ..., faça:
1.1 Simule M_1 com entrada w por i passos.
1.2 Simule M_2 com entrada w por i passos.
1.3 Se M_1 ou M_2 aceitam, ACEITE."

■

b. Intersecção

Vamos construir uma M.T. M_4 que reconhece a linguagem $L_4 = L_1 \cap L_2$.

M_4 = "Com entrada w:
1. Para i = 1, 2, 3, ..., faça:
1.1 Simule M_1 com entrada w por i passos.
1.2 Simule M_2 com entrada w por i passos.
Se M_1 e M_2 aceitam, ACEITE."

■

c. Concatenação

Vamos construir uma M.T. M_5 que reconhece a linguagem $L_5 = L_1 \bullet L_2$.

M_5 = "Com entrada $w = w_1w_2w_3...w_n$, onde $n = |w|$ e w_i é cada um dos caracteres de w:
1. Para k = 1, 2, 3, ..., faça:
1.1 Para i de 0 até n, faça:
1.1.1 Simule M_1 com entrada $w_1w_2...w_i$ por k passos.
1.1.2 Simule M_2 com entrada $w_{i+1}...w_n$ por k passos.
Se M_1 e M_2 aceitam, ACEITE."

■

d. Estrela

Vamos construir uma M.T. M_6 que reconhece a linguagem $L_6 = L_1^*$.

M_6 = "Com entrada $w = w_1w_2w_3...w_n$, onde $n = |w|$ e w_i é cada um dos caracteres de w:
1. Se $w = \varepsilon$, ACEITE.
2. Para k = 1, 2, 3, ..., faça:
2.1 Para i de 0 até n faça:
2.1.1 Simule M_1 com entrada $w_1w_2...w_i$ por k passos.
2.1.2 Se M_1 aceita, então:
2.1.2.1 Simule M_6 com entrada $w_{i+1}...w_n$ por k passos.
2.1.2.2 Se M_6 aceita, ACEITE."

■

....

Exercício 6 [3.16 Sipser] (!) Mostre que uma linguagem é recursiva se e somente se algum enumerador enumera a linguagem em ordem lexicográfica.

RESPOSTA =

($\rightarrow$)

Seja L_1 uma linguagem recursiva. Logo, existe uma M.T. M que decide L_1, tal que $L(M) = L_1$.

Considere $\sum^* = \{s_1, s_2, s_3, ...\}$ uma lista de todas as palavras formadas pelo alfabeto $\sum$ em ordem lexicográfica.

Seja E um enumerador.

E = "Ignore a entrada:

1. Repita para i = 1, 2, 3, ...

1.1 Simule M para cada entrada $s_1, s_2, s_3, ..., s_i$, seqüência esta já descrita .

1.2 Se alguma computação aceita, imprima o correspondente s_i."

($\leftarrow$)

Seja L_2 uma linguagem tal que existe um enumerador E, onde $L(E) = L_2$.

Seja uma M.T. N com entrada w.

Seja E' um enumerador, onde L(E') = L' e $L' = L_2 \cup \{u \mid u$ é uma palavra maior que $w\}$ e E' imprime palavras em ordem lexicográfica.

Vamos construir N.

N = "Com entrada w:

1. Simule E'. Cada vez que E' imprime uma palavra w', compare-a com w.
2. Se w = w', ACEITE.
3. Se w < w', REJEITE."

Logo, N decide L_2. Assim, L_2 é recursiva.

■

....

Exercício 7 Prove pelo método da diagonalização que o conjunto das linguagens sobre o alfabeto {0,1} não é enumerável. Justifique todos os passos.

RESPOSTA =

Seja £ o conjunto das linguagens sobre o alfabeto $\sum = \{0, 1\}$.

Vamos tomar L_i Є £ tal que se uma palavra w sobre $\sum$ pertence à linguagem L_i, coloca-se um 1 na i-ésima posição da palavra B. Caso contrário, coloque um 0.

A palavra B é a seqüência característica de L. A posição em que irá se colocar o 0 ou o 1 é a posição de w em ordem lexicográfica de $\sum^*$.

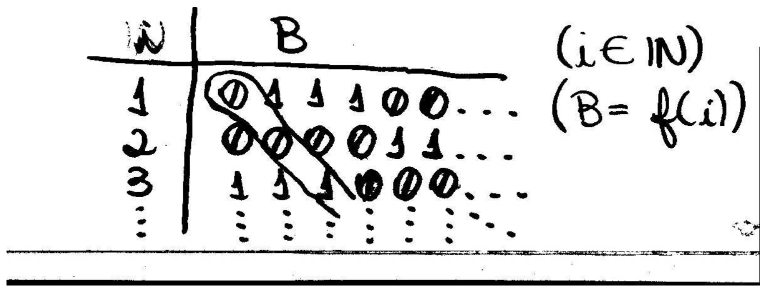
Ex.: $\sum^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, ...\}$

L = {ε, 1, 00, 01, 10, ...}

B = 1 0 1 1 1 1 0 0 ...

Como B é infinita, vamos provar pelo método da diagonalização que B não é enumerável.

Sabemos que existe a função $f: B \rightarrow L$ e $f^{-1}: L \rightarrow B$.



Vamos construir um <u>x</u> tal que f(i) ≠ x, para todo i.
Seja $x = d_1d_2d_3...$. Escolha para d_i um dígito diferente do i-ésimo dígito de f(i).
Então: x será diferente de f(1) no 1º dígito.
x será diferente de f(2) no 2º dígito.
x será diferente de f(3) no 3º dígito.
...
x será diferente de f(n) no nº dígito.

Portanto, x não está na imagem de f. Contradição.

■

....