

Seja $T = (Q, A, T, \delta, i, f, \Delta)$ uma máquina de Turing. Uma configuração $c = (q, utv)$ de T diz-se uma configuração de:

paragem se δ não está definida em (q, t) ou, $u = e$ e $\delta(q, t) = (q', t', E)$ para alguns $q' \in Q$ e $t' \in T$;

aceitação se $q = f$;

rejeição se c é uma configuração de paragem e $q \neq f$;

ciclo se a partir de c não é possível computar uma configuração de paragem.

Sejam $k \in \mathbb{N}$, A e T alfabetos com $A \subseteq T$ e $g : (A^*)^k \rightarrow T^*$ uma função parcial em k variáveis. Diz-se que g é uma função **Turing-computável** se existe uma máquina de Turing, de alfabeto de entrada A e alfabeto de fita contendo T , que calcula g .

Uma linguagem L diz-se:

recursivamente enumerável se existe uma MT que reconhece L ;

recursiva (ou decidível) se existe uma MT que decide L .

Todas as linguagens recursivas são recursivamente enumeráveis.

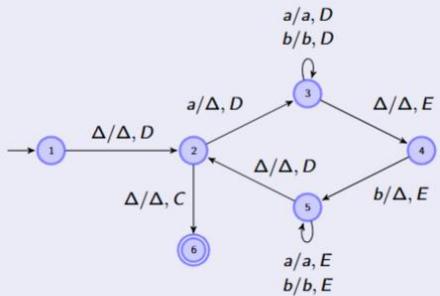
Seja L uma linguagem sobre um alfabeto A . A **função característica de L** é a função

$$\chi_L : A^* \rightarrow \{0, 1\}$$

definida, para cada $u \in A^*$, por

$$\chi_L(u) = \begin{cases} 1 & \text{se } u \in L \\ 0 & \text{se } u \notin L \end{cases}$$

O diagrama



descreve uma MT cujo alfabeto de entrada contém as letras a e b . Como veremos, esta MT reconhece a linguagem não regular $\{a^n b^n \mid n \in \mathbb{N}_0\}$.

• Funções numéricas podem também ser descritas por máquinas de Turing desde que se consiga representar os números através de palavras.

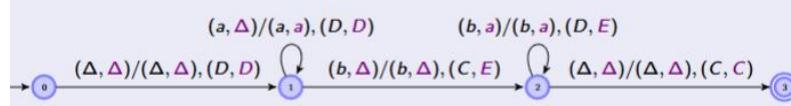
• Para funções envolvendo o conjunto \mathbb{N}_0 dos inteiros não negativos, utilizaremos a representação "unária" no alfabeto $\{1\}$, em que

$$n \in \mathbb{N} \text{ é representado pela palavra } 1^n = \underbrace{1 \cdots 1}_{n \text{ vezes}}.$$

Ou seja, \mathbb{N}_0 é identificado com $\{1\}^*$.

Uma linguagem $L \subseteq A^*$ é **recursiva** se e só se existe uma máquina de Turing \mathcal{T} (dita um **algoritmo**) que reconhece L e que pára sempre que parte da configuração inicial de uma palavra $u \in L$ (ou seja, nenhuma configuração inicial é de ciclo).

A figura abaixo representa uma MT \mathcal{T} sobre $\{a, b\}$ com 2 fitas



que aceita a linguagem $\{a^n b^n \mid n \in \mathbb{N}\}$.

Sejam L e K linguagens sobre um alfabeto A .

- i) Se L e K são **recursivas** (resp. **recursivamente enumeráveis**), então $L \cup K$ e $L \cap K$ são **recursivas** (resp. **recursivamente enumeráveis**).
- ii) Se L é **recursiva**, então \bar{L} é **recursiva**.

Demonstração: A ideia em i) é considerar máquinas de Turing \mathcal{T}_L e \mathcal{T}_K que **decidam** (resp. **aceitem**) L e K , respetivamente, e construir uma máquina de Turing com conjunto de estados $Q_L \times Q_K$ (onde Q_L e Q_K são os conjuntos de estados de \mathcal{T}_L e \mathcal{T}_K respetivamente) e com **duas fitas**, que executa em simultâneo segundo \mathcal{T}_L e \mathcal{T}_K .

DEFINIÇÃO [FUNÇÃO CODIFICADORA]

Define-se uma função injetiva

$$\begin{aligned} c : MT_N &\longrightarrow \{x, y\}^* \\ \mathcal{T} &\longmapsto c(\mathcal{T}) \end{aligned}$$

que associa a cada máquina de Turing \mathcal{T} (normalizada) uma palavra $c(\mathcal{T})$ sobre o alfabeto $\{x, y\}$, chamada "o **código** de \mathcal{T} ", da forma descrita a seguir.

Começa por associar-se uma palavra $c'(\cdot)$, sobre o alfabeto $\{x\}$,

- a cada $q_i \in Q$, $s_i \in S$ e aos movimentos C, E, D :

$$\begin{aligned} c'(q_i) &= c'(s_i) = x^{j+1}, \\ c'(C) &= x, \quad c'(E) = x^2, \quad c'(D) = x^3. \end{aligned}$$

Note-se que, em particular,

$$c'(\Delta) = c'(s_0) = x \quad \text{e} \quad c'(f) = c'(q_0) = x.$$

Depois, codifica-se a máquina de Turing \mathcal{T} pela palavra

$$c(\mathcal{T}) = c'(q_1)y c'(e_1)y c'(e_2) \cdots y c'(e_k)y$$

onde q_1 é o estado inicial de \mathcal{T} e e_1, e_2, \dots, e_k são as transições de \mathcal{T} numa ordem fixa previamente.

Pode também codificar-se cada palavra $w = r_1 r_2 \cdots r_n$, onde $r_i \in S$, por

$$c(w) = yy c'(r_1) y c'(r_2) \cdots y c'(r_n) y.$$

Seja $A = \{a, b\}$ e seja

$$\text{conc} : A^* \times A^* \rightarrow A^*$$

a função de concatenação de duas palavras de A^* . Uma MT, $\mathcal{T}_{\text{conc}}$, que calcula esta função é obtida utilizando a máquina $\mathcal{T}_{\text{apagar}}$, que apaga a letra da célula apontada pelo cursor. Pretende-se que $\mathcal{T}_{\text{conc}}$ realize a computação

$$(i, \Delta u \Delta v) \xrightarrow{*} (f, \Delta uv)$$

para cada $u, v \in A^*$. A ideia é que a máquina $\mathcal{T}_{\text{conc}}$, apresentada abaixo,



apague o símbolo branco que separa as palavras u e v . Logo a função de concatenação sobre o alfabeto $\{a, b\}$ é **Turing-computável**. Em geral, prova-se que a função de concatenação sobre qualquer alfabeto é **Turing-computável**.

- Consideremos a função

$$+ : \mathbb{N}_0 \times \mathbb{N}_0 \rightarrow \mathbb{N}_0$$

de adição em \mathbb{N}_0 .

- A seguinte máquina de Turing calcula $+$



onde $\mathcal{T}_{\text{apagar}}$ é a máquina de Turing que apaga a letra lida pelo cursor.

Uma linguagem L é **recursiva** se e só se L e \bar{L} são **recursivamente enumeráveis**.

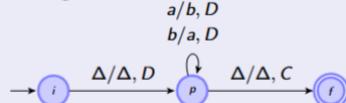
Seja A um alfabeto e sejam:

- $RE(A)$ o conjunto das linguagens **recursivamente enumeráveis** sobre A ;
- $MT(A)$ o conjunto das máquinas de Turing (normalizadas) de alfabeto de entrada A .

O conjunto $MT(A)$ é **numerável**. Logo $RE(A)$ também é **numerável** pois cada linguagem de $RE(A)$ é reconhecida por uma máquina de Turing de $MT(A)$.

O teorema seguinte mostra que as linguagens **recursivamente enumeráveis** formam uma ínfima parte do conjunto de todas as linguagens.

Seja \mathcal{T} a máquina de Turing



que percorre uma dada palavra sobre o alfabeto $\{a, b\}$ trocando cada a por b e vice-versa. Suponhamos, para simplificar, que

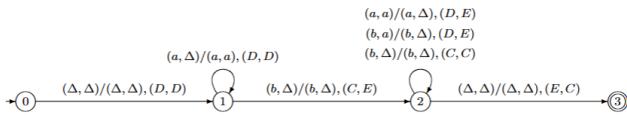
$$i = q_1, \quad p = q_2 \quad (\text{e } f = q_0), \quad a = s_1, \quad b = s_2 \quad (\text{e } \Delta = s_0),$$

e que as 4 transições são tomadas na ordem em que aparecem no grafo (da esquerda para a direita e de cima para baixo).

A 1ª transição e_1 , descrita por $\delta(i, \Delta) = (p, \Delta, D)$, i.e., $\delta(q_1, s_0) = (q_2, s_0, D)$ é codificada por $c'(e_1) = x^2 y x y x^3 y x y x^3 y$, enquanto que \mathcal{T} é codificada por

$$c(\mathcal{T}) = \underbrace{x^2}_{c'(q_1)} \underbrace{y x y x y x^3 y}_{c'(e_1)} \underbrace{x^3 y x^2 y x^3 y}_{c'(e_2)} \underbrace{x^3 y x^2 y x^3 y}_{c'(e_3)} \underbrace{x^3 y x y x y}_{c'(e_4)}$$

2. Seja $A = \{a, b\}$ e seja T a seguinte máquina de Turing sobre A com duas fitas,



- a) Indique a sequência de configurações que podem ser computadas a partir da configuração $(0, \underline{\Delta}aaabab, \underline{\Delta})$ e diga se a palavra $aaabab$ é aceite por T .

R: A sequência de configurações de T que podem ser computadas a partir da configuração $(0, \underline{\Delta}aaabab, \underline{\Delta})$ é a seguinte:

$$(0, \underline{\Delta}aaabab, \underline{\Delta}) \longrightarrow (1, \underline{\Delta}aaabab, \underline{\Delta\Delta}) \xrightarrow{*} (1, \underline{\Delta}aaabab, \underline{\Delta}aaa\underline{\Delta}) \longrightarrow (2, \underline{\Delta}aaabab, \underline{\Delta}aaa) \\ \longrightarrow (2, \underline{\Delta}aaabab, \underline{\Delta}aa) \longrightarrow (2, \underline{\Delta}aaabab, \underline{\Delta}a) \longrightarrow (2, \underline{\Delta}aaabab\underline{\Delta}, \underline{\Delta}) \longrightarrow (3, \underline{\Delta}aaabab, \underline{\Delta}).$$

A palavra $aaabab$ é portanto aceite por T , já que, a partir da configuração inicial desta palavra se computou uma configuração de aceitação, isto é, uma configuração cujo estado é o estado final de T (neste caso o estado 3).

- b) Identifique a linguagem L reconhecida por T .

R: A linguagem reconhecida por T é

$$L = \{a^n bx : x \in A^*, n = |bx|\}.$$

Com efeito, partindo da configuração inicial de uma palavra $w \in A^*$, para se atingir o estado final da máquina T é necessário passar do estado 1 para o estado 2. Para isso tem que se ler uma (a primeira) ocorrência de b em w , depois de ter lido todas as ocorrências anteriores da letra a e de copiar para a 2ª fita. Então w tem de ser da forma $w = a^n bx$ para alguns $n \in \mathbb{N}_0$ e $x \in A^*$. Agora, no estado 2, por cada letra da palavra bx que a máquina T lê na 1ª fita, apaga um a na 2ª fita. Portanto, os cursorres atingem simultaneamente o símbolo branco em cada fita se e só se $n = |bx|$, e só nesse caso é que T vai para o estado final.

- c) Para que palavras $u \in A^*$, $(0, \underline{\Delta}u, \underline{\Delta})$ é uma configuração de ciclo?

R: Como se pode verificar, analisando o comportamento da máquina T , uma configuração $(0, \underline{\Delta}u, \underline{\Delta})$ é de ciclo se e só se a partir dela se chega a uma configuração em que a máquina está no estado 2, o cursor da 1ª fita está a ler b e o cursor da 2ª fita está a ler Δ (e está na 1ª célula). Ou seja, se é possível fazer uma computação

$$(0, \underline{\Delta}u, \underline{\Delta}) \xrightarrow{*} (2, u_1 \underline{bu}_2, \underline{\Delta})$$

em T . Note-se para além disso que, neste caso, $u = u_1 bu_2$ pois o conteúdo da 1ª fita nunca é alterado. A palavra u_1 pode tomar duas formas:

- $u_1 = \epsilon$. Neste caso, $u = bu_2 \in bA^*$ e chega-se à configuração $(2, u_1 \underline{bu}_2, \underline{\Delta})$ logo que o estado 2 é atingido, ou seja, imediatamente depois de ler o b inicial da palavra u .
- $u_1 = a^n bx$ para alguns $n \in \mathbb{N}$ e $x \in A^*$, com $n = |bx|$. Neste caso, $u_1 \in L$ (onde $u \in LbA^*$) e chega-se à configuração $(2, u_1 \underline{bu}_2, \underline{\Delta})$ porque depois de se esgotarem os n a 's na 2ª fita e de se ter avançado o factor bx em u , ainda “sobrou” em u o sufixo bu_2 , começado pela letra b .

Em resumo, $(0, \underline{\Delta}u, \underline{\Delta})$ é uma configuração de ciclo se e só se

$$u \in bA^* \cup LbA^*.$$

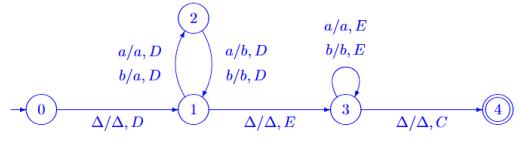
- d) Para que palavras $v \in A^*$, a partir de $(0, \underline{\Delta}v, \underline{\Delta})$ pode ser computada uma configuração de rejeição?

R: Pode ser computada uma configuração de rejeição a partir de $(0, \underline{\Delta}v, \underline{\Delta})$ se e só se esta configuração não é de ciclo e a partir dela não pode ser computada uma configuração de aceitação, ou seja, se e só se

$$v \in A^* \setminus (bA^* \cup LbA^* \cup L) = a^* \cup LaA^*.$$

Note-se que, nos casos em que $v \in a^*$, a máquina T pára no estado 1, e quando $v \in LaA^*$ a máquina T pára no estado 2 (depois de ter lido o prefixo $a^n bx \in L$ de u e de ter ficado a ler a letra a que sucede a esse prefixo).

R: T pode ser representada graficamente da seguinte forma:



- b) Indique a sequência de configurações que podem ser computadas a partir da configuração $(0, \underline{\Delta}aaaabaab)$.

R: A sequência de configurações de T que podem ser computadas a partir da configuração $(0, \underline{\Delta}aaaabaab)$ é a seguinte:

$$(0, \underline{\Delta}aaaabaab) \longrightarrow (1, \underline{\Delta}aaaabaab) \longrightarrow (2, \underline{\Delta}agaabaab) \longrightarrow (1, \underline{\Delta}abgabaab) \\ \longrightarrow (2, \underline{\Delta}ababaab) \longrightarrow (1, \underline{\Delta}ababbaab) \longrightarrow (2, \underline{\Delta}ababqab) \longrightarrow (1, \underline{\Delta}abababab) \\ \longrightarrow (2, \underline{\Delta}abababab) \longrightarrow (1, \underline{\Delta}abababab\underline{\Delta}) \longrightarrow (3, \underline{\Delta}abababab) \xrightarrow{*} (3, \underline{\Delta}abababab) \\ \longrightarrow (4, \underline{\Delta}abababab).$$

- c) Identifique o domínio D da função g .

R: O domínio D da função g é a linguagem reconhecida por T . Ora, partindo da configuração inicial $(0, \underline{\Delta}u)$ de uma palavra $u \in A^*$, a máquina T chega ao estado final se e só se u tem comprimento par (quando u tem comprimento ímpar, T pára no estado 2). Logo,

$$D = L(T) = \{u \in A^* : |u| = 2k, k \in \mathbb{N}_0\}.$$

- d) Para cada elemento $u \in D$, determine a palavra $g(u)$.

R: Para $u \in D$, tem-se

$$g(u) = (ab)^{\frac{|u|}{2}}.$$

De facto, T substitui, em u : cada letra numa posição ímpar por a (quando transita do estado 1 para o estado 2); cada letra numa posição par por b (quando transita do estado 2 para o estado 1). Dado que o comprimento de $|u|$ é par, o resultado é portanto o indicado.