MAC 414
Autômatos, Computabilidade e Complexidade
aula 11 — 21/10/2020

Máquina de Turing

Máquina de Turing

Baseada no **L&P**. **Sipser** e **Rich** definem diferente.

Máquina de Turing

Baseada no **L&P**. **Sipser** e **Rich** definem diferente.

Uma Máquina de Turing é uma quíntupla $(K, \Sigma, \delta, s, H)$, onde

- K é o conjunto finito de estados.
- Σ é o alfabeto. Símbolo especial: $\sqcup$ (espaço) ; símbolos $\rightarrow$ e $\leftarrow$ e $\triangleright$ (início) não estão em Σ.
- $s \in K$ é o estado inicial.
- $H \subseteq K$ é o conjunto de estados de parada.
- A função de transição
  $\delta : K \setminus H \times \Sigma \cup \{\triangleright\} \rightarrow K \times \Sigma \cup \{\leftarrow, \rightarrow\}$
  satisfazendo $\delta(p, \triangleright) = (q, \rightarrow)$.

Significado

Uma MT tem uma fita de entrada e uma cabeça de leitura que aponta uma posição por vez. A transição depende do estado atual e do que é lido na fita; a primeira componente indica o novo estado:

- Se $\delta(p, a) = (q, b)$, escreve b nessa posição.

Significado

Uma MT tem uma fita de entrada e uma cabeça de leitura que aponta uma posição por vez. A transição depende do estado atual e do que é lido na fita; a primeira componente indica o novo estado:

- Se $\delta(p, a) = (q, b)$, escreve b nessa posição.
- Se $\delta(p, a) = (q, \leftarrow)$, anda para a esquerda uma posição.

Significado

Uma MT tem uma fita de entrada e uma cabeça de leitura que aponta uma posição por vez. A transição depende do estado atual e do que é lido na fita; a primeira componente indica o novo estado:

- Se $\delta(p, a) = (q, b)$, escreve b nessa posição.
- Se $\delta(p, a) = (q, \leftarrow)$, anda para a esquerda uma posição.
- Se $\delta(p, a) = (q, \rightarrow)$, anda para a direita uma posição.

Significado

Uma MT tem uma fita de entrada e uma cabeça de leitura que aponta uma posição por vez. A transição depende do estado atual e do que é lido na fita; a primeira componente indica o novo estado:

- Se $\delta(p, a) = (q, b)$, escreve b nessa posição.
- Se $\delta(p, a) = (q, \leftarrow)$, anda para a esquerda uma posição.
- Se $\delta(p, a) = (q, \rightarrow)$, anda para a direita uma posição.

  O início $\triangleright$ empurra a leitora para a direita.

Formalização

Formalização

Uma configuração de M é um elemento de

$$K \times \triangleright\Sigma^* \times (\Sigma^* \setminus \Sigma^* \sqcup).$$

Formalização

Uma configuração de M é um elemento de

$$K \times \triangleright\Sigma^* \times (\Sigma^* \setminus \Sigma^* \sqcup).$$

(p, u, v) significa estado p, fita uv, cabeça de leitura na primeira letra de v.

Formalização

Uma configuração de M é um elemento de

$$K \times \triangleright\Sigma^* \times (\Sigma^* \setminus \Sigma^* \sqcup).$$

(p, u, v) significa estado p, fita uv, cabeça de leitura na primeira letra de v.

Notação:

$(q, \triangleright a, aba) \longrightarrow (q, \triangleright a\underline{a}ba)$
$(h, \triangleright \sqcup\sqcup, \sqcup a) \longrightarrow (h, \triangleright \sqcup\sqcup\underline{\sqcup}a)$
$(q, \triangleright \sqcup\sqcup a\sqcup, \lambda) \longrightarrow (q, \triangleright \sqcup\sqcup a\sqcup\underline{\sqcup})$

Computação

Computação

Computação em um passo

$$(p, w_1 \underline{a}_1 w_1) \vdash_M (q, w_2 \underline{a}_2 w_2)$$

se existe $b \in \Sigma \cup \{\rightarrow, \leftarrow\}$ tal que $\delta(p, a_1) = (q, b)$ e

1. $b \in \Sigma, w_2 = w_1, u_2 = u_1$ e $a_2 = b$, ou

Computação

Computação em um passo

$$(p, w_1 \underline{a}_1 w_1) \vdash_M (q, w_2 \underline{a}_2 w_2)$$

se existe $b \in \Sigma \cup \{\rightarrow, \leftarrow\}$ tal que $\delta(p, a_1) = (q, b)$ e

1. $b \in \Sigma, w_2 = w_1, u_2 = u_1$ e $a_2 = b$, ou
2. $b = \leftarrow, w_1 = w_2 a_2$ e
   a) $u_2 = a_1 u_1$, se $a_1 \neq \sqcup$ ou $u_1 \neq \lambda$, ou
   b) $u_2 = \lambda$, se $a_1 = \sqcup$ e $u_1 = \lambda$, ou

Computação

Computação em um passo

$$(p, w_1 \underline{a_1} u_1) \vdash_M (q, w_2 \underline{a_2} u_2)$$

se existe $b \in \Sigma \cup \{\rightarrow, \leftarrow\}$ tal que $\delta(p, a_1) = (q, b)$ e

1. $b \in \Sigma, w_2 = w_1, u_2 = u_1$ e $a_2 = b$, ou
2. $b = \leftarrow, w_1 = w_2 a_2$ e
   a) $u_2 = a_1 u_1$, se $a_1 \neq \sqcup$ ou $u_1 \neq \lambda$, ou
   b) $u_2 = \lambda$, se $a_1 = \sqcup$ e $u_1 = \lambda$, ou
3. $b = \rightarrow, w_2 = w_1 a_1$ e
   a) $u_1 = a_2 u_2$, ou
   b) $u_1 = u_2 = \lambda$ e $a_2 = \sqcup$.

Computação

Computação em um passo

$$(p, w_1 \underline{a}_1 w_1) \vdash_M (q, w_2 \underline{a}_2 w_2)$$

se existe $b \in \Sigma \cup \{\rightarrow, \leftarrow\}$ tal que $\delta(p, a_1) = (q, b)$ e

1. $b \in \Sigma, w_2 = w_1, u_2 = u_1$ e $a_2 = b$, ou
2. $b = \leftarrow, w_1 = w_2 a_2$ e
   a) $u_2 = a_1 u_1$, se $a_1 \neq \sqcup$ ou $u_1 \neq \lambda$, ou
   b) $u_2 = \lambda$, se $a_1 = \sqcup$ e $u_1 = \lambda$, ou
3. $b = \rightarrow, w_2 = w_1 a_1$ e
   a) $u_1 = a_2 u_2$, ou
   b) $u_1 = u_2 = \lambda$ e $a_2 = \sqcup$.

Computação

Computação em um passo

$$(p, w_1 \underline{a_1} u_1) \vdash_M (q, w_2 \underline{a_2} u_2)$$

se existe $b \in \Sigma \cup \{\rightarrow, \leftarrow\}$ tal que $\delta(p, a_1) = (q, b)$ e

1. $b \in \Sigma, w_2 = w_1, u_2 = u_1$ e $a_2 = b$, ou
2. $b = \leftarrow, w_1 = w_2 a_2$ e
   a) $u_2 = a_1 u_1$, se $a_1 \neq \sqcup$ ou $u_1 \neq \lambda$, ou
   b) $u_2 = \lambda$, se $a_1 = \sqcup$ e $u_1 = \lambda$, ou
3. $b = \rightarrow, w_2 = w_1 a_1$ e
   a) $u_1 = a_2 u_2$, ou
   b) $u_1 = u_2 = \lambda$ e $a_2 = \sqcup$.

$\vdash_M^*$ é o fecho reflexivo transitivo de $\vdash_M$.

Dados para uma MT

Dados para uma MT

Alfabeto de entrada $\Sigma_0 \subseteq \Sigma \setminus \{\sqcup\}$

Dados para uma MT

Alfabeto de entrada $\Sigma_0 \subseteq \Sigma \setminus \{\sqcup\}$

Entrada para M é $w \in \Sigma_0^*$: (M, w) se lê
"M com entrada w".

Dados para uma MT

Alfabeto de entrada $\Sigma_0 \subseteq \Sigma \setminus \{\sqcup\}$

Entrada para M é $w \in \Sigma_0^*$: (M, w) se lê
"M com entrada w".

Convenção: (M, w) significa configuração inicial

$$(s, \triangleright \underline{\sqcup} w).$$

Linguagem gráfica para MTs

Linguagem gráfica para MTs

Convenções:

1. Cada MT tem um único estado de parada.

Linguagem gráfica para MTs

Convenções:

1. Cada MT tem um único estado de parada.
2. As várias MTs aparecendo num diagrama têm conjuntos de estados disjuntos.

Linguagem gráfica para MTs

Convenções:

1. Cada MT tem um único estado de parada.
2. As várias MTs aparecendo num diagrama têm conjuntos de estados disjuntos.
3. Uma flecha rotulada $M_1 \longrightarrow M_2$ significa uma coleção de transições do estado final h_1 de M_1 para o estado inicial s_2 de M_2, da forma $\delta(h_1, \sigma) = (s_2, \sigma)$.

Linguagem gráfica para MTs

Convenções:

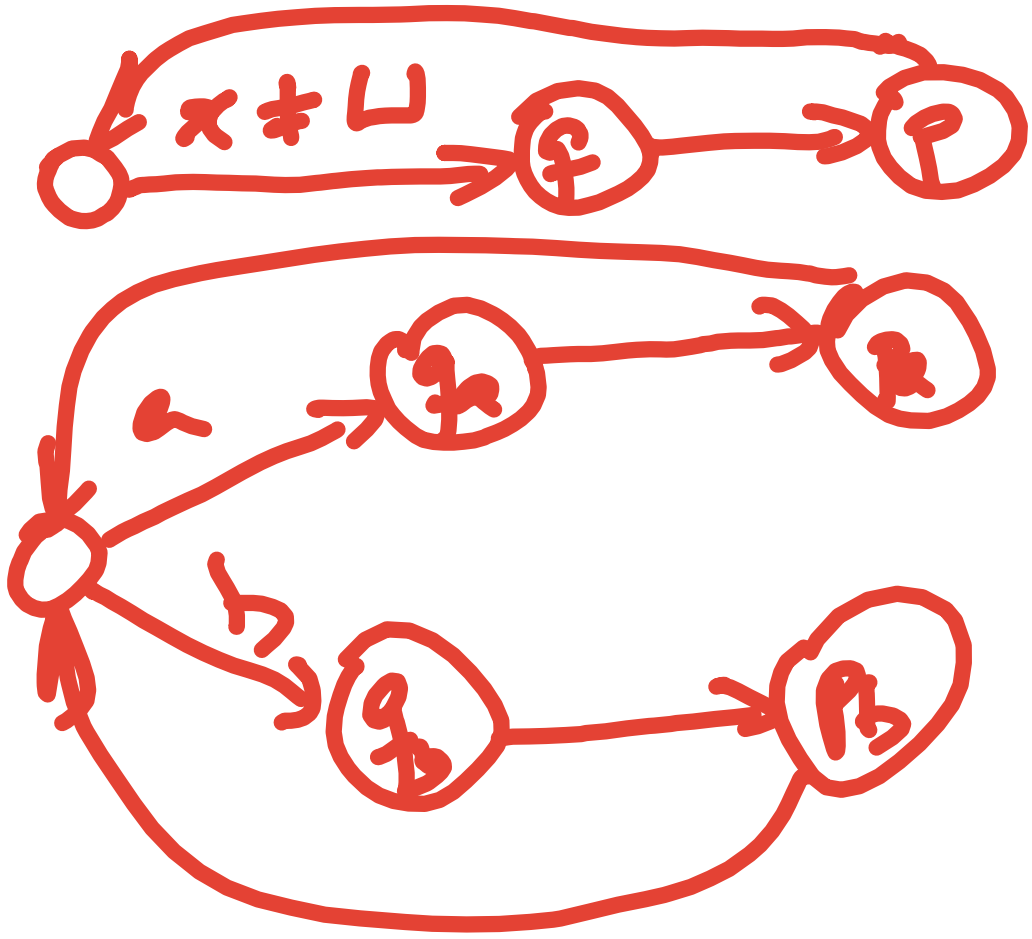
1. Cada MT tem um único estado de parada.
2. As várias MTs aparecendo num diagrama têm conjuntos de estados disjuntos.
3. Uma flecha rotulada $M_1 \longrightarrow M_2$ significa uma coleção de transições do estado final h_1 de M_1 para o estado inicial s_2 de M_2, da forma $\delta(h_1, \sigma) = (s_2, \sigma)$.
    1. Se o rótulo é um conjunto de caracteres, uma transição por rótulo. Açúcar sintático: $\bar{\sigma} = \Sigma \backslash \{\sigma\}$, nada=$\Sigma$.

Linguagem gráfica para MTs

Convenções:

1. Cada MT tem um único estado de parada.
2. As várias MTs aparecendo num diagrama têm conjuntos de estados disjuntos.
3. Uma flecha rotulada $M_1 \longrightarrow M_2$ significa uma coleção de transições do estado final h_1 de M_1 para o estado inicial s_2 de M_2, da forma $\delta(h_1, \sigma) = (s_2, \sigma)$.
    1. Se o rótulo é um conjunto de caracteres, uma transição por rótulo. Açúcar sintático: $\bar{\sigma} = \Sigma \backslash \{\sigma\}$, nada=$\Sigma$.
    2. Rótulo da forma $a \in A$, ou $a \neq b$ indicam que a deve ser lembrado; para cada estado seguinte deve haver uma cópia para cada valor possível de a.

$\Sigma = \{a, b, \sqcup\}$



Máquinas úteis

- M_a: dada por $\delta(s, \sigma) = (h, a)$.

Máquinas úteis

- M_a: dada por $\delta(s, \sigma) = (h, a)$.
- R: dada por $\delta(s, \sigma) = (h, \rightarrow)$.

Máquinas úteis

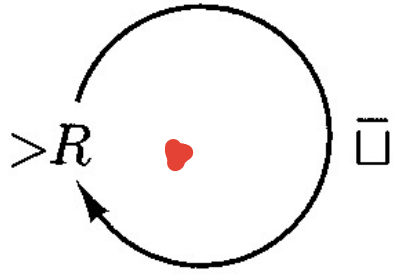
- M_a: dada por $\delta(s, \sigma) = (h, a)$.
- R: dada por $\delta(s, \sigma) = (h, \rightarrow)$.
- L: dada por $\delta(s, \sigma) = (h, \leftarrow)$.

Máquinas úteis

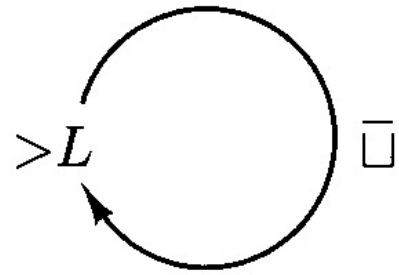
- M_a: dada por $\delta(s, \sigma) = (h, a)$.
- R: dada por $\delta(s, \sigma) = (h, \rightarrow)$.
- L: dada por $\delta(s, \sigma) = (h, \leftarrow)$.

  Convenção: $R \longrightarrow R \longrightarrow M_a \longrightarrow M_b \longrightarrow L$ denotado simplesmente por $RRabL$.

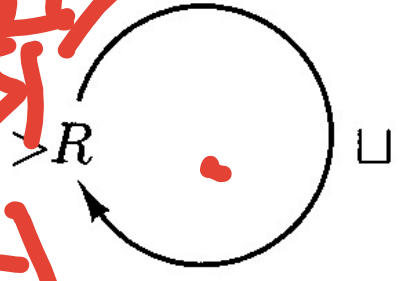
Máquinas úteis



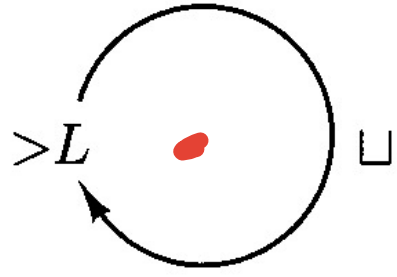
(a) $R_{\sqcup}$



(b) $L_{\sqcup}$



(c) $R_{\bar{\sqcup}}$



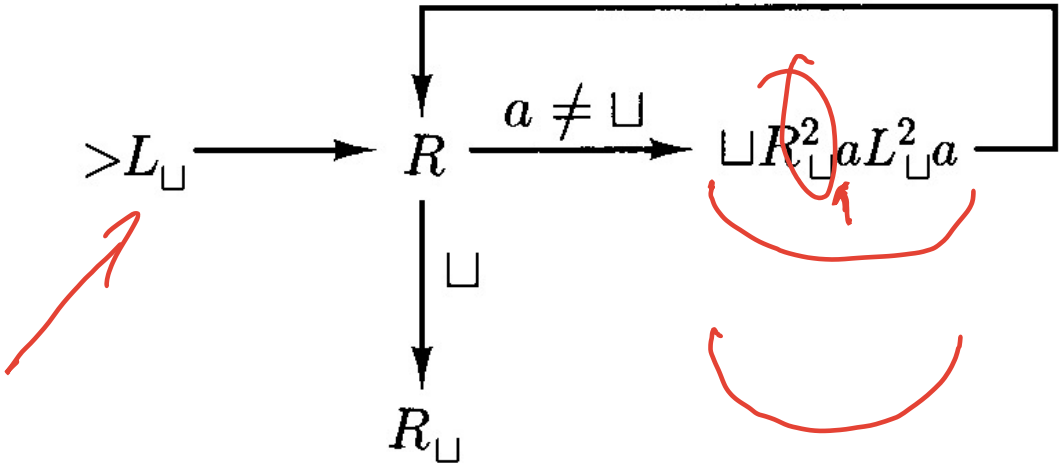
(d) $L_{\bar{\sqcup}}$

Exemplo: copiadora

Dado: $\triangleright \sqcup w \underline{\sqcup}$

Produz: $\triangleright \sqcup w \sqcup w \underline{\sqcup}$

C:

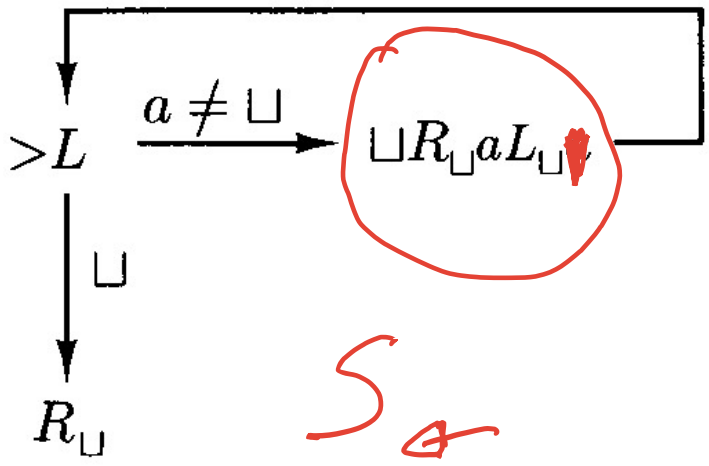


Exemplo: shift right

Dado: $\triangleright \sqcup w \underline{\sqcup}$

Produz: $\triangleright \sqcup \sqcup w \underline{\sqcup}$

$S_{\rightarrow}$:



Decidir linguagens

Decidir linguagens

Decisor: $H = \{a, r\}$

Decidir linguagens

Decisor: $H = \{a, r\}$

M aceita w se (M, w) para em a.

Decidir linguagens

Decisor: $H = \{a, r\}$

M aceita w se (M, w) para em a.

M rejeita w se (M, w) para em r.

Decidir linguagens

Decisor: $H = \{a, r\}$

M aceita w se (M, w) para em a.

M rejeita w se (M, w) para em r.

M decide $L \subseteq \Sigma_0^*$ sse
Se $w \in L$, M aceita w, se $w \notin L$, M rejeita w.

Decidir linguagens

Decisor: $H = \{a, r\}$

M aceita w se (M, w) para em a.

M rejeita w se (M, w) para em r.

M decide $L \subseteq \Sigma_0^*$ sse
Se $w \in L$, M aceita w, se $w \notin L$, M rejeita w.

L é recursiva se alguma máquina de Turing decide L.

Decidir linguagens

Decisor: $H = \{a, r\}$

M aceita w se (M, w) para em a.

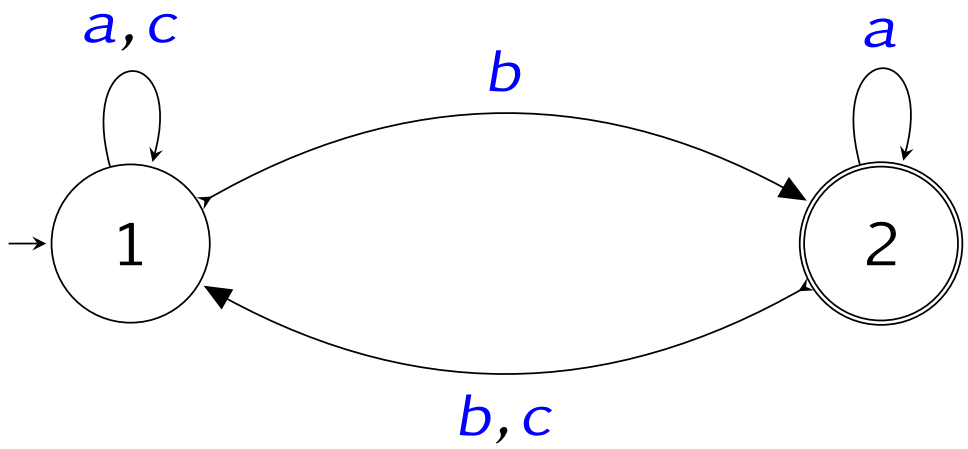
M rejeita w se (M, w) para em r.

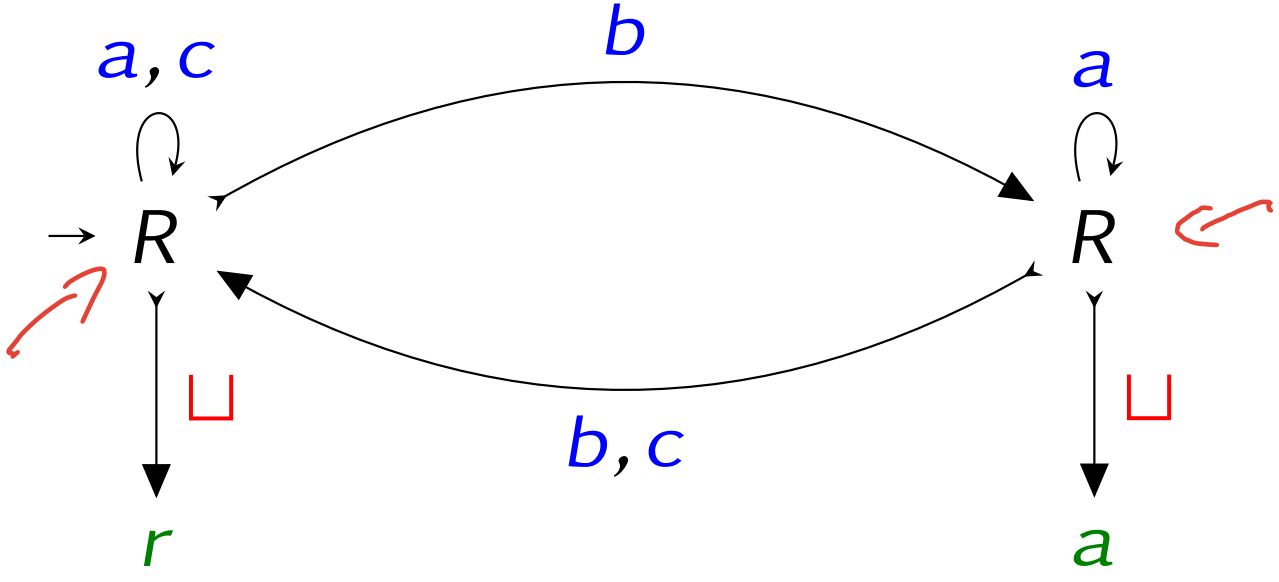
M decide $L \subseteq \Sigma_0^*$ sse
Se $w \in L$, M aceita w, se $w \notin L$, M rejeita w.

L é recursiva se alguma máquina de Turing decide L.

Prop: Se L é recursiva, seu complemento também é.

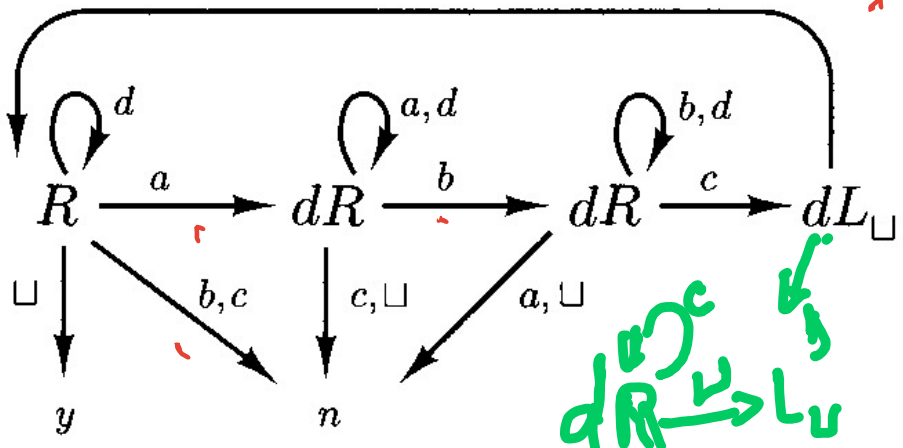
Exemplo: AD

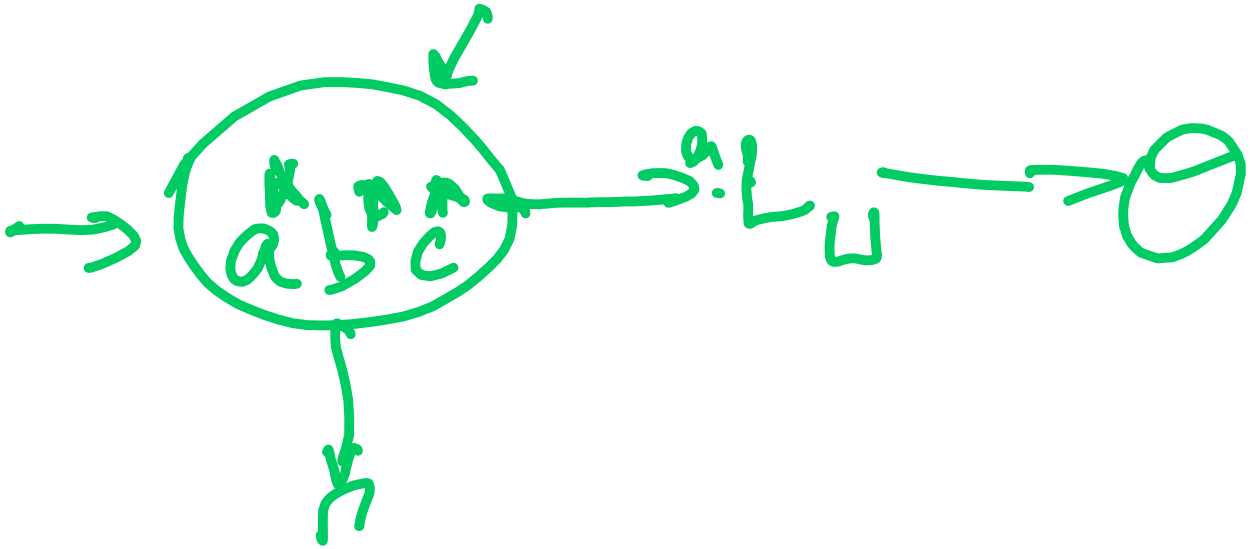




Exemplo: $a^n b^n c^n$

Figura copiada: $a \to y$, $r \to n$




a
b
c
a: [ ]

Funções computáveis

Funções computáveis

M com um único estado final h, $\Sigma_0 \subset \Sigma$.

Funções computáveis

M com um único estado final h, $\Sigma_0 \subset \Sigma$.

Suponha (M, w) para e existe u tal que $(s, \triangleright \underline{\sqcup} w) \vdash_M^* (h, \triangleright \underline{\sqcup} u)$. Então u é a saída de M com entrada w, e escrevemos $u = M(w)$.

Funções computáveis

M com um único estado final h, $\Sigma_0 \subset \Sigma$.

Suponha (M, w) para e existe u tal que $(s, \triangleright \underline{\sqcup} w) \vdash_M^* (h, \triangleright \underline{\sqcup} u)$. Então u é a saída de M com entrada w, e escrevemos $u = M(w)$.

Se para cada entrada M tem uma saída, então $w \mapsto M(w)$ é uma função, a função computada por M.

Funções computáveis

M com um único estado final h, $\Sigma_0 \subset \Sigma$.

Suponha (M, w) para e existe u tal que $(s, \triangleright \underline{\sqcup} w) \vdash_M^* (h, \triangleright \underline{\sqcup} u)$. Então u é a saída de M com entrada w, e escrevemos $u = M(w)$.

Se para cada entrada M tem uma saída, então $w \mapsto M(w)$ é uma função, a função computada por M.

Uma função é recursiva se existe uma MT que a computa.

Prop: Se f, g são funções computáveis, então $f \circ g$ também é.

$(f \circ g)(w) := f(g(w))$

Dem: $M_g \longrightarrow M_f$

$M_g M_f$

Ligação com linguagens

$L \subseteq \Sigma^*$, sua função característica é $\chi_L : \Sigma^* \to \{0,1\}$ é dada por

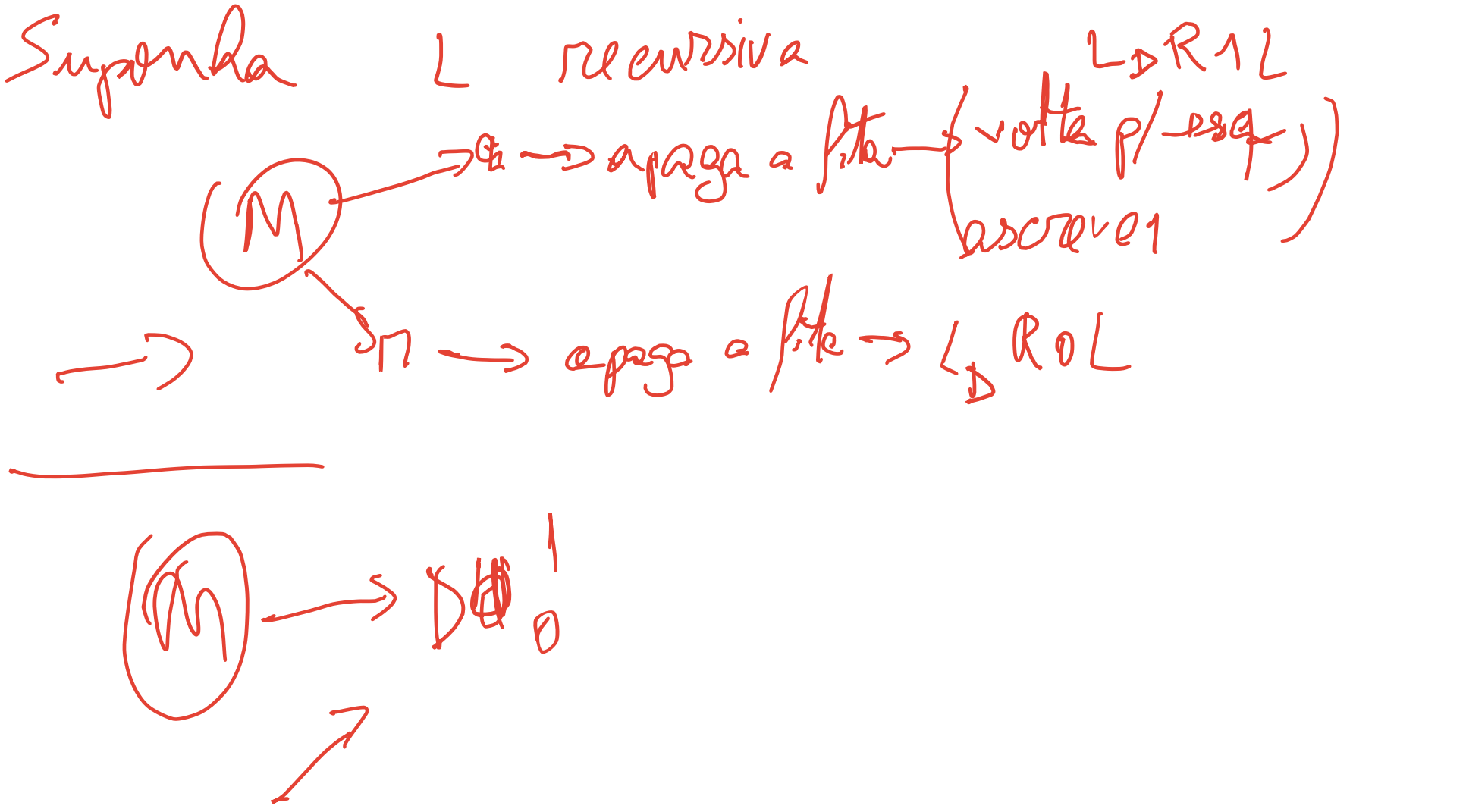
$$\chi_L(w) = \begin{cases} 1, & w \in L \\ 0, & w \notin L \end{cases}$$

Ligação com linguagens

$L \subseteq \Sigma^*$, sua função característica é $\chi_L : \Sigma^* \to \{0,1\}$ é dada por

$$\chi_L(w) = \begin{cases} 1, & w \in L \\ 0, & w \notin L \end{cases}$$

Prop: L é recursiva sse sua função caracteristica é recursiva.


Suponha L recursiva
L ▷ R 1 L
M
q1 → apaga a fita → (volta p/ esq, escreve 1)
q0 → apaga a fita → L ▷ R 0 L
M
B B B 1
0

Funções numéricas

Fixando uma representação para os naturais (ou para os inteiros), palavras sobre os dígitos podem ser interpretadas como números. Assim, MTs podem computar funções numéricas

Funções numéricas

Fixando uma representação para os naturais (ou para os inteiros), palavras sobre os dígitos podem ser interpretadas como números. Assim, MTs podem computar funções numéricas

Juntando um sinal de pontuação, pode-se representar tuplas de inteiros.

Funções numéricas

Fixando uma representação para os naturais (ou para os inteiros), palavras sobre os dígitos podem ser interpretadas como números. Assim, MTs podem computar funções numéricas

Juntando um sinal de pontuação, pode-se representar tuplas de inteiros.

A definição de função recursiva se estende naturalmente a outros domínios, além de palavras.

Recursivamente enumerável

Considere uma MT com alfabeto de entrada Σ_0^*. Então M semidecide a linguagem $\{w \in \Sigma_0^* \mid (M, w) \text{ para}\}$.

Recursivamente enumerável

Considere uma MT com alfabeto de entrada Σ_0^*. Então M semidecide a linguagem $\{w \in \Sigma_0^* \mid (M, w) \text{ para}\}$.

L é recursivamente enumerável se existe uma MT que a semidecide.

Recursivamente enumerável

Considere uma MT com alfabeto de entrada Σ_0^*. Então M semidecide a linguagem $\{w \in \Sigma_0^* \mid (M, w) \text{ para}\}$.

L é recursivamente enumerável se existe uma MT que a semidecide.

$M(w) = \nearrow$ se (M, w) não para.

Recursivamente enumerável

Considere uma MT com alfabeto de entrada Σ_0^*. Então M semidecide a linguagem $\{w \in \Sigma_0^* \mid (M, w) \text{ para}\}$.

L é recursivamente enumerável se existe uma MT que a semidecide.

$M(w) = \nearrow$ se (M, w) não para.

Prop: Toda linguagem recursiva é recursivamente enumerável.