Licenciatura em Ciências da Computação

COMPUTABILIDADE E COMPLEXIDADE

1. Máquinas de Turing

José Carlos Costa

Dep. Matemática
Universidade do Minho
Braga, Portugal

1º semestre 2021/2022

Definição

Uma *máquina de Turing* é um septeto $\mathcal{T} = (Q, A, T, \delta, i, f, \Delta)$:

1. Q é um conjunto finito não vazio, dito o *conjunto de estados* de $\mathcal{T}$;
2. A é um alfabeto, chamado o *alfabeto (de entrada)* de $\mathcal{T}$;
3. $T \supseteq A$ é um alfabeto, dito o *alfabeto da fita* de $\mathcal{T}$, tal que $Q \cap T = \emptyset$. O conjunto $T \setminus A$ é chamado o *alfabeto auxiliar* de $\mathcal{T}$;
4. $\delta : Q \times T \to Q \times T \times \{E, C, D\}$ é uma função parcial, dita a *função transição* de $\mathcal{T}$, indefinida em (f, t) para cada $t \in T$.
   Os *movimentos* são:
   E-esquerda; D-direita; C-"centro" (ausência de movimento);
5. $i \in Q$, dito o *estado inicial* de $\mathcal{T}$;
6. $f \in Q$, chamado o *estado final* de $\mathcal{T}$;
7. $\Delta \in T \setminus A$ é um símbolo auxiliar, designado o *símbolo branco*.

Exemplo

O septeto

$$\mathcal{T} = (\{1,2,3\}, \{a,b\}, \{a,b,\Delta\}, \delta, 1, 3, \Delta)$$

onde δ é a função parcial de $\{1,2,3\} \times \{a,b\}$ em $\{1,2,3\} \times \{a,b\} \times \{E,C,D\}$ tal que

$$\begin{array}{rclcrcl} \delta(1,\Delta) & = & (2,\Delta,D) & & \delta(2,a) & = & (2,a,D) \\ \delta(2,b) & = & (2,b,D) & & \delta(2,\Delta) & = & (3,\Delta,C) \end{array}$$

é uma máquina de Turing com:

- três estados (1, 2 e 3);
- alfabeto de entrada $\{a,b\}$;
- Δ como único símbolo auxiliar;
- 1 e 3 como estados inicial e final.

Exemplo (continuação)

A função transição δ que, recorde-se, é definida por

$$\begin{aligned} \delta(1,\Delta) &= (2,\Delta,D) & \qquad \delta(2,a) &= (2,a,D) \\ \delta(2,b) &= (2,b,D) & \qquad \delta(2,\Delta) &= (3,\Delta,C) \end{aligned}$$

pode ser representada pela tabela

δ	a	b	Δ
1			$(2,\Delta,D)$
2	$(2,a,D)$	$(2,b,D)$	$(3,\Delta,C)$

Na tabela omite-se a linha referente ao estado final pois, por definição, numa máquina de Turing não existem transições a sair do estado final.

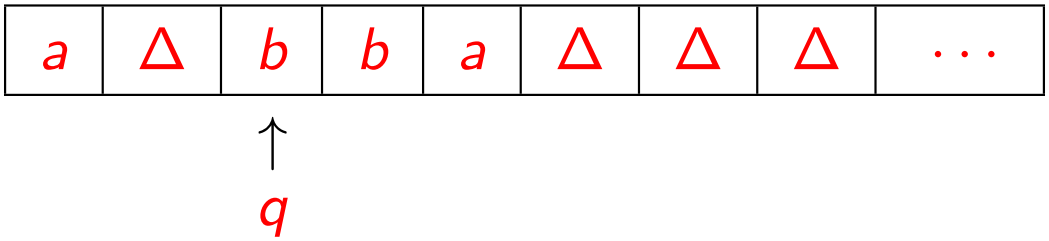
Uma máquina de Turing

$$\mathcal{T} = (Q, A, T, \delta, i, f, \Delta)$$

é dotada de uma:

1. **fita**
   - dividida em células;
   - infinita à direita;
   - com uma célula inicial à esquerda;
   - cada célula tem uma letra de T;
   - em cada instante o número de células não brancas (ou seja células onde não está escrito o símbolo branco Δ) é finito;
2. **cabeça** ou **cursor** (de leitura e escrita)
   - posicionada em cada momento numa determinada célula da fita;
   - permite ler a letra da célula e substituir essa letra por outra (eventualmente a mesma).

Por exemplo, a figura



representa a fita e a cabeça de uma máquina de Turing em que:

- nas 1ª e 5ª células da fita está escrita a letra a;
- nas 3ª e 4ª células da fita está escrito b;
- todas as outras células da fita estão em branco;
- a seta indica que a cabeça está posicionada na 3ª célula;
- a letra q por baixo da seta indica o estado atual da máquina de Turing.

A cabeça pode efetuar movimentos determinados pela função transição. Assim, a igualdade

$$\delta(q, t) = (q', t', m)$$

significa que, se

- a máquina está no estado q;
- a cabeça está posicionada numa célula da fita onde está escrita a letra t;

então

- a máquina transita para o estado q';
- substitui a letra t pela letra t' na fita;
- a cabeça efetua o movimento m (ou seja, move-se para a esquerda, para a direita ou não se move, conforme m seja E, D ou C).

Exemplo

Seja

$$\mathcal{T} = (\{1,2,3\}, \{a,b\}, \{a,b,\Delta\}, \delta, 1, 3, \Delta)$$

a máquina de Turing cuja função transição δ é dada pela tabela

δ	a	b	Δ
1			$(2, \Delta, D)$
2	$(2, a, D)$	$(2, a, D)$	$(3, \Delta, C)$

Num dado momento, uma situação possível de fita e de cursor é

Δ	a	a	a	b	Δ	Δ	$\cdots$

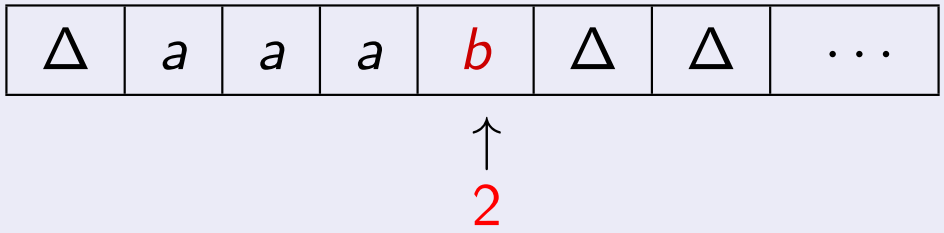
$\uparrow$

2

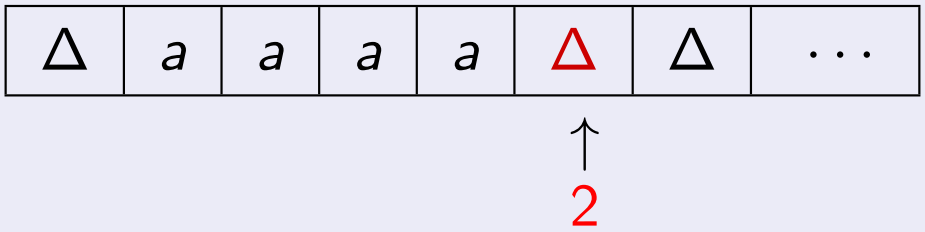
que indica que a máquina está posicionada no estado 2 e o cursor está posicionado na 4ª célula onde está escrita a letra a.

Exemplo (continuação)

Dado que $\delta(2, a) = (2, a, D)$, no momento seguinte, a situação de fita e de cursor passará a ser



ou seja, apenas mudou a posição do cursor que se situa agora na 5ª célula.
Dado que $\delta(2, b) = (2, a, D)$, a situação posterior seria então



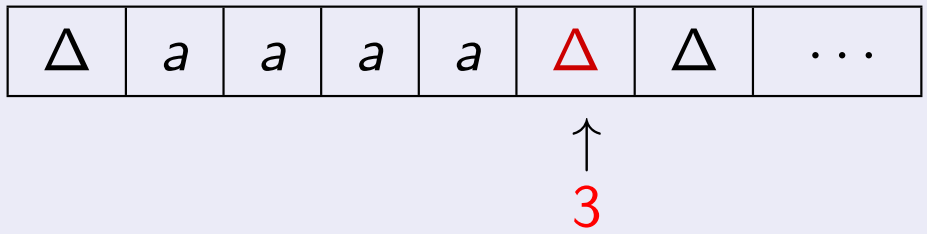
Note-se que neste passo houve uma alteração da letra na 5ª célula da fita mantendo-se ainda a máquina no estado 2.

Exemplo (continuação)

No próximo passo, usando a igualdade

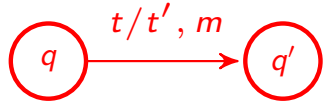
$$\delta(2, \Delta) = (3, \Delta, C)$$

obtém-se



Como a máquina atingiu o estado 3, que é o estado final da máquina, não é possível efetuar mais passos.

A representação de máquinas de Turing é feita de forma análoga aos autómatos finitos e aos autómatos de pilha. A única diferença é na representação das transições:

$\delta(q, t) = (q', t', m)$ é representada por 

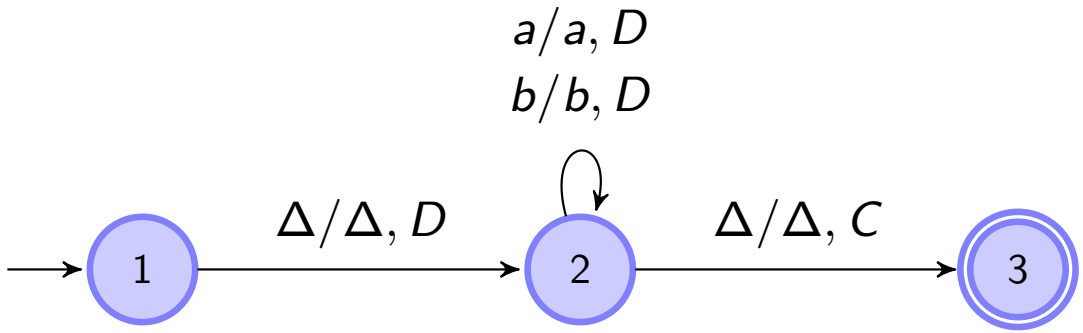
e significa, conforme referido anteriormente, que, se

- a máquina está no estado q;
- o cursor está a ler o símbolo t na fita;

então

- a máquina transita para o estado q';
- na fita, na posição do cursor, a letra t é substituída por t';
- o cursor efetua o movimento m.

Por exemplo, o diagrama



representa a máquina de Turing do exemplo da página 3. Esta máquina não é ainda fundamentalmente diferente de um autómato finito pois:

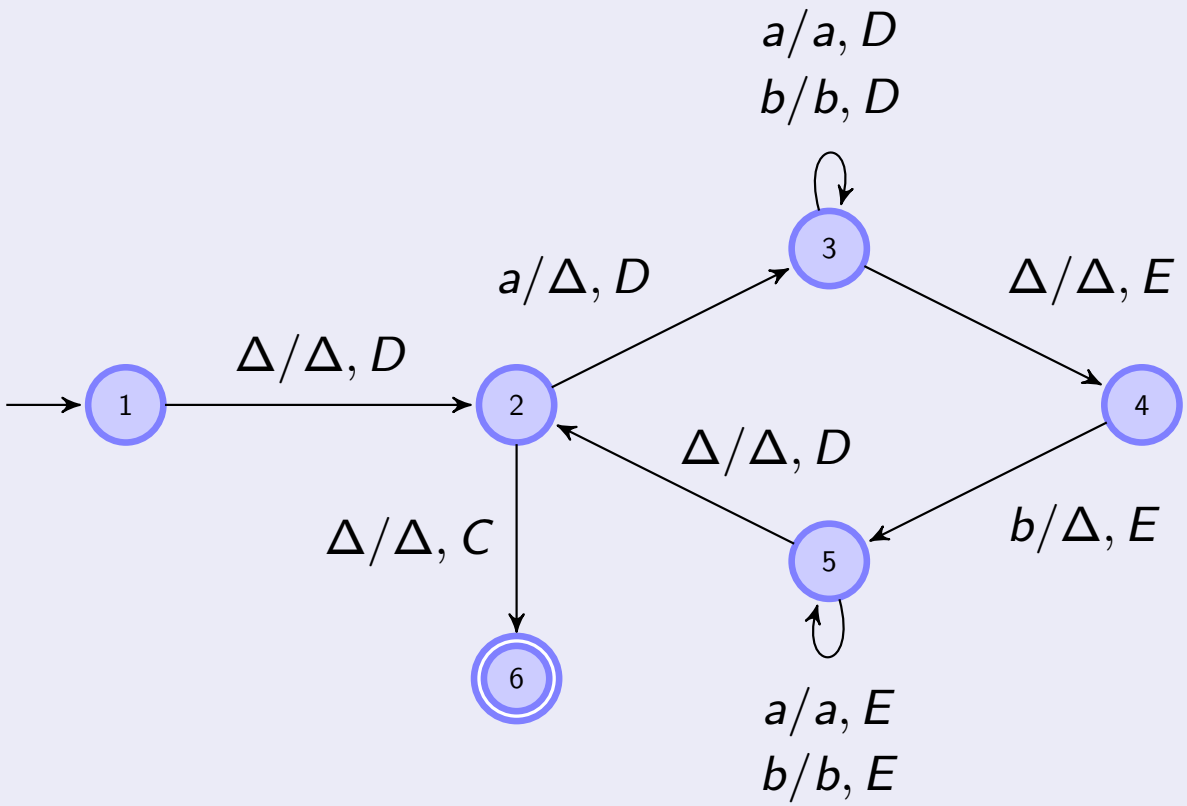
- não substitui letras na fita por outras diferentes;
- com exceção da transição para o estado final, o único movimento que faz é para a direita.

Veremos mais tarde que, de facto, esta máquina reconhece uma linguagem regular, ou seja, reconhece uma linguagem reconhecida por autómato finito.

Uma máquina de Turing mais "sofisticada" é apresentada a seguir.

Exemplo

O diagrama



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

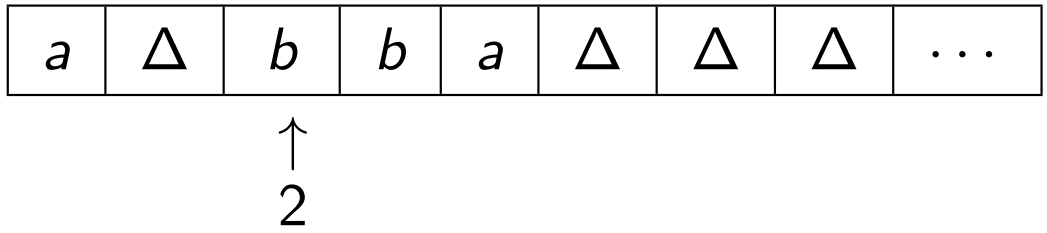
Definição

Seja $\mathcal{T} = (Q, A, T, \delta, i, f, \Delta)$ uma máquina de Turing. Uma *configuração* de $\mathcal{T}$ é um par ordenado $(q, u\underline{t}v)$ onde $q \in Q$, $u, v \in T^*$, $t \in T$ e:

1. q é o estado atual da máquina;
2. a palavra utv está escrita na fita o mais à esquerda possível (i.e., a 1ª letra de utv ocupa a 1ª célula) e todas as células após v estão preenchidas com Δ;
3. a cabeça está posicionada na célula ocupada pela letra t.

- Uma *configuração* representa a situação da fita e do cursor da MT num dado momento.
- Por isso, a configuração $(q, u\underline{t}v)$ pode ser também denotada por $(q, u\underline{t}v\Delta^n)$ para cada $n \in \mathbb{N}$.

- Por exemplo, a configuração $(2, a\Delta\underline{b}bba)$ representa a situação



- Esta configuração pode também ser denotada por $(2, a\Delta\underline{b}bba\Delta\Delta)$.

Notação

Para uma palavra $v \in T^*$, utilizaremos a notação $(q, u\underline{v})$, onde $q \in Q$ e $u \in T^*$, para representar a configuração:

- $(q, u\underline{t}v')$ se $t \in T$ e $v' \in T^*$ são tais que $v = tv'$;
- $(q, u\underline{\Delta})$ se $v = \epsilon$.

- Assim, no exemplo acima a configuração $(2, a\Delta\underline{b}bba)$ pode ainda ser representada por $(2, a\Delta\underline{bba})$, por $(2, a\Delta\underline{bba\Delta})$, etc.

Definição

Sejam $c_1 = (q_1, u_1 \underline{t_1} v_1)$ e $c_2 = (q_2, u_2 \underline{t_2} v_2)$ duas configurações de $\mathcal{T}$.
Diz-se que:

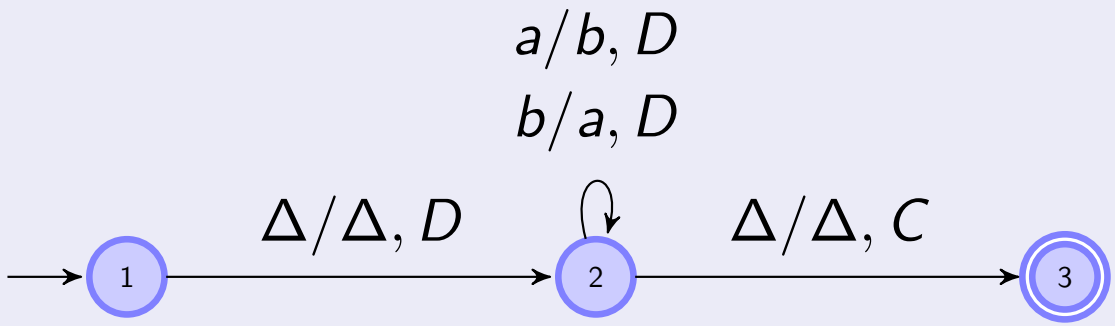
- c_2 é uma *computação direta* a partir de c_1, e escreve-se $c_1 \xrightarrow[\mathcal{T}]{} c_2$, se $\mathcal{T}$ passa da configuração c_1 para a configuração c_2 *num único passo*.

  [Assim, no exemplo das páginas 8-10, a configuração $(2, \Delta aaa\underline{b})$ é uma computação direta a partir de $(2, \Delta aa\underline{a}b)$ e poderíamos escrever $(2, \Delta aa\underline{a}b) \xrightarrow[\mathcal{T}]{} (2, \Delta aaa\underline{b})$.]
- c_2 é uma *computação* a partir de c_1, e escreve-se $c_1 \xrightarrow[\mathcal{T}]{*} c_2$, se $\mathcal{T}$ passa da configuração c_1 para a configuração c_2 *em zero ou mais passos*; ou seja, se $c_1 = c_2$ ou $c_1 = c'_1 \xrightarrow[\mathcal{T}]{} c'_2 \xrightarrow[\mathcal{T}]{} c'_3 \cdots c'_{n-1} \xrightarrow[\mathcal{T}]{} c'_n = c_2$ para configurações $c'_1, c'_2, \ldots, c'_n$.

Quando não houver ambiguidade em relação à máquina de Turing $\mathcal{T}$ que se está a considerar, simplificaremos as notações de $\xrightarrow[\mathcal{T}]{}$ e de $\xrightarrow[\mathcal{T}]{*}$ omitindo a letra $\mathcal{T}$.

Exemplo

Seja $\mathcal{T}$ a seguinte máquina de Turing



A partir da configuração $(1, \underline{\Delta}aaaba)$ podem ser efetuadas em $\mathcal{T}$, sucessivamente, as seguintes computações diretas

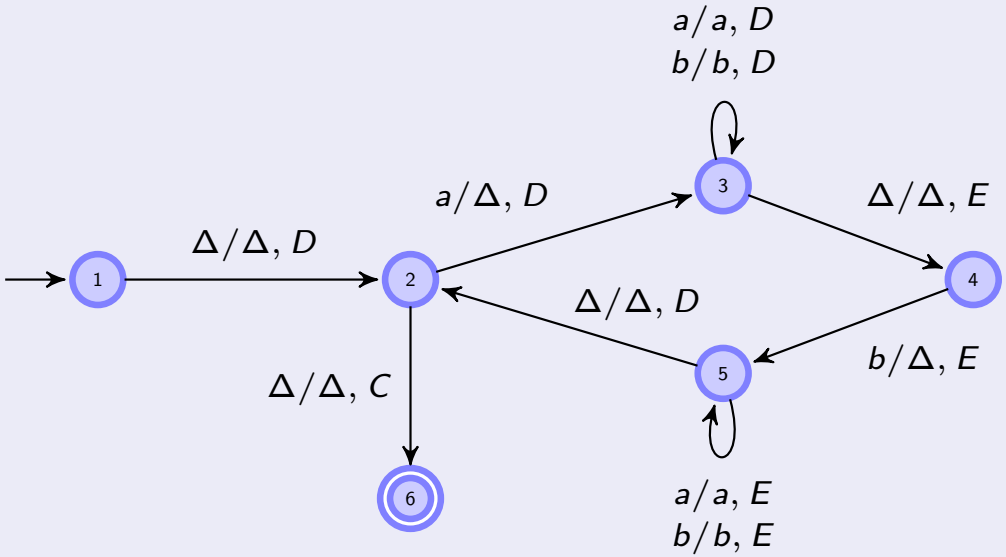
$$(1, \underline{\Delta}aaaba) \longrightarrow (2, \Delta\underline{a}aaba) \longrightarrow (2, \Delta b\underline{a}aba) \longrightarrow (2, \Delta bb\underline{a}ba) \longrightarrow$$

$$(2, \Delta bbb\underline{b}a) \longrightarrow (2, \Delta bbba\underline{a}) \longrightarrow (2, \Delta bbbab\underline{\Delta}) \longrightarrow (3, \Delta bbbab\underline{\Delta}).$$

Portanto $(1, \underline{\Delta}aaaba) \xrightarrow{*} (3, \Delta bbbab\underline{\Delta})$ é uma computação em $\mathcal{T}$.

Exercício

Seja $\mathcal{T}$ a máquina de Turing (do exemplo da página 13) representada pelo grafo



Indique a sequência de configurações que podem ser computadas em $\mathcal{T}$ a partir da configuração: (i) $(1, \underline{\Delta}abb)$; (ii) $(1, \underline{\Delta}aabb)$.

(i) $(1, \underline{\Delta}abb) \longrightarrow (2, \Delta\underline{a}bb) \longrightarrow (3, \Delta\Delta\underline{b}b) \longrightarrow (3, \Delta\Delta b\underline{b}) \longrightarrow (3, \Delta\Delta bb\underline{\Delta})$
$\longrightarrow (4, \Delta\Delta b\underline{b}) \longrightarrow (5, \Delta\Delta\underline{b}) \longrightarrow (5, \Delta\underline{\Delta}b) \longrightarrow (2, \Delta\Delta\underline{b})$.

(ii) $(1, \underline{\Delta}aabb) \longrightarrow (2, \Delta\underline{a}abb) \longrightarrow (3, \Delta\Delta\underline{a}bb) \overset{*}{\longrightarrow} (3, \Delta\Delta abb\underline{\Delta}) \longrightarrow (4, \Delta\Delta ab\underline{b})$
$\longrightarrow (5, \Delta\Delta a\underline{b}) \overset{*}{\longrightarrow} (5, \Delta\underline{\Delta}ab) \longrightarrow (2, \Delta\Delta\underline{a}b) \longrightarrow (3, \Delta^3\underline{b}) \longrightarrow$
$(3, \Delta^3 b\underline{\Delta}) \longrightarrow (4, \Delta^3\underline{b}) \longrightarrow (5, \Delta^2\underline{\Delta}) \longrightarrow (2, \Delta^3\underline{\Delta}) \longrightarrow (6, \Delta^3\underline{\Delta})$.

Definição

Seja $\mathcal{T} = (Q, A, T, \delta, i, f, \Delta)$ uma máquina de Turing. Uma configuração

$$c = (q, u\underline{t}v)$$

de $\mathcal{T}$ diz-se uma *configuração de*:

- *paragem* se δ não está definida em (q, t) ou, $u = \epsilon$ 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.

Diz-se ainda que:

- $(i, \underline{\Delta}w)$ é a *configuração inicial* associada a uma palavra $w \in A^*$;

Definição (continuação)

- uma *palavra* $w \in A^*$ é *aceite* por $\mathcal{T}$ se existe uma computação de uma configuração de aceitação a partir da configuração inicial associada a w. Ou seja, se existe uma computação

$$(i, \underline{\Delta} w) \xrightarrow[\mathcal{T}]{*} (f, u\underline{t}v)$$

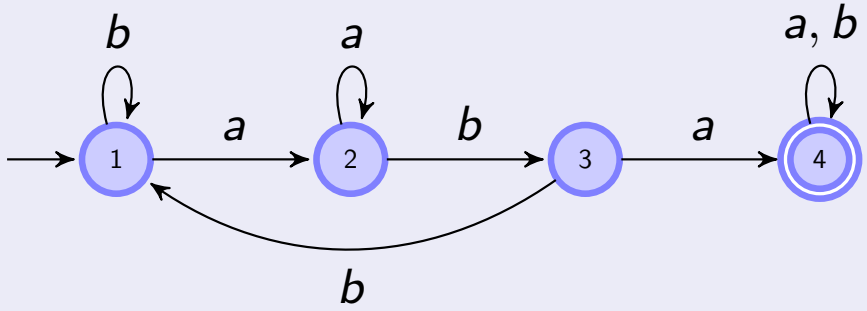
  para uma certa configuração de aceitação $(f, u\underline{t}v)$.
- a *linguagem aceite* ou *reconhecida* por $\mathcal{T}$, representada por $L(\mathcal{T})$, é o conjunto das palavras aceites por $\mathcal{T}$.

Note-se que uma palavra w poderá não ser aceite por uma de duas razões: partindo da configuração inicial associada a w,

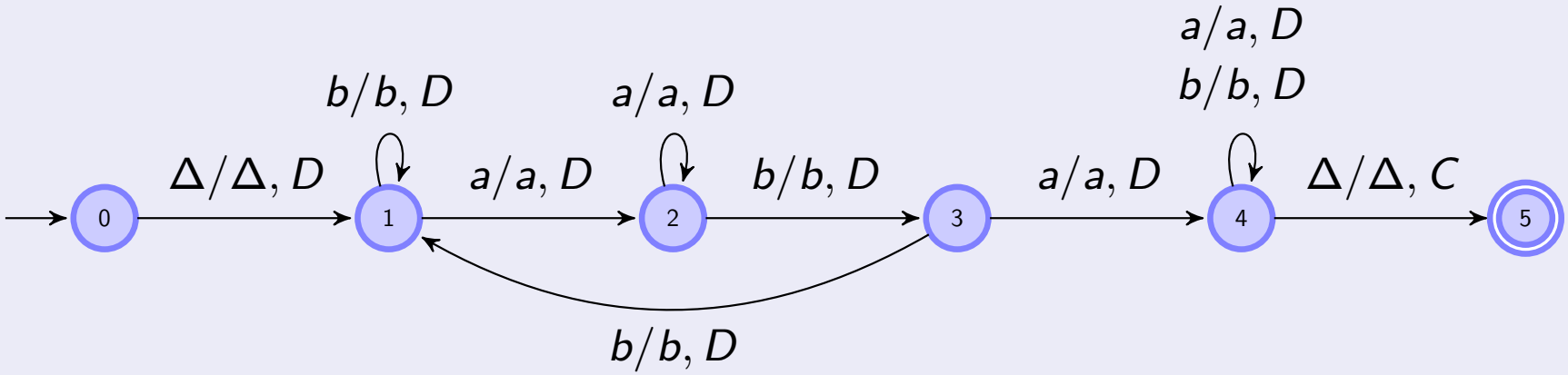
- a máquina de Turing pára numa configuração de rejeição;
- a máquina de Turing não pára (ou seja, a configuração inicial associada a w é uma configuração de ciclo).

Exemplo

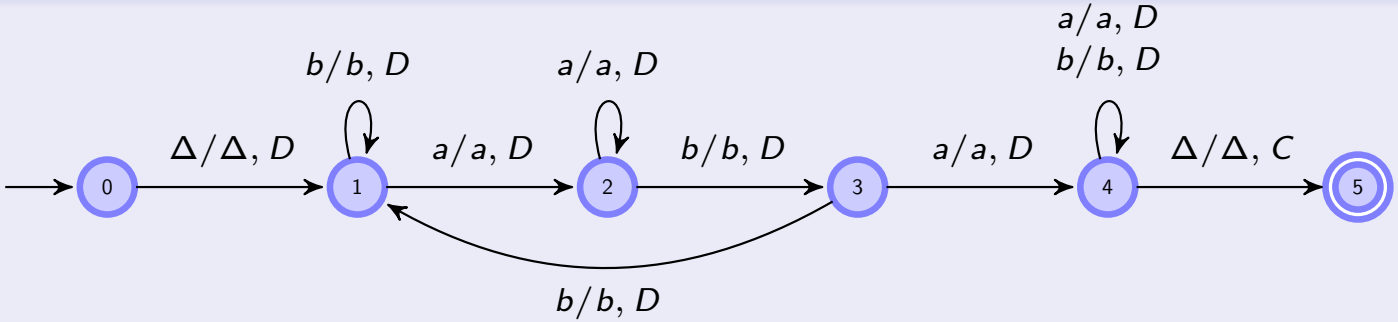
Mostraremos neste exemplo como se pode construir uma máquina de Turing que reconhece a linguagem regular $L = A^*abaA^*$ sobre o alfabeto $A = \{a, b\}$. Um autómato finito determinista que reconhece L é



Então, pode-se verificar que a seguinte máquina de Turing $\mathcal{T}$ reconhece L



Exemplo (continuação)



- Verifiquemos que $\mathcal{T}$ aceita a palavra *aaba*. Em $\mathcal{T}$ pode-se computar

$$(0, \underline{\Delta}aaba) \longrightarrow (1, \Delta\underline{a}aba) \longrightarrow (2, \Delta a\underline{a}ba) \longrightarrow (2, \Delta aa\underline{b}a) \longrightarrow$$
$$(3, \Delta aab\underline{a}) \longrightarrow (4, \Delta aaba\underline{\Delta}) \longrightarrow (5, \Delta aaba\underline{\Delta}).$$

  Dado que 5 é o estado final de $\mathcal{T}$, a configuração $(5, \Delta aaba\underline{\Delta})$ é uma configuração de aceitação. Logo a palavra *aaba* é aceite por $\mathcal{T}$.

- Pelo contrário, $\mathcal{T}$ não aceita a palavra *abba*. De facto,

$$(0, \underline{\Delta}abba) \longrightarrow (1, \Delta\underline{a}bba) \longrightarrow (2, \Delta a\underline{b}ba) \longrightarrow$$
$$(3, \Delta ab\underline{b}a) \longrightarrow (1, \Delta abb\underline{a}) \longrightarrow (2, \Delta abba\underline{\Delta}).$$

  Ou seja, atingiu-se uma configuração de rejeição pois a máquina parou num estado que não é o final e, portanto, a palavra *abba* não é aceite por $\mathcal{T}$.

A construção do exemplo anterior pode ser generalizada para todas as linguagens regulares, o que mostra que a classe de linguagens reconhecidas por máquinas de Turing contém a classe das linguagens reconhecidas por autómatos finitos.

Teorema

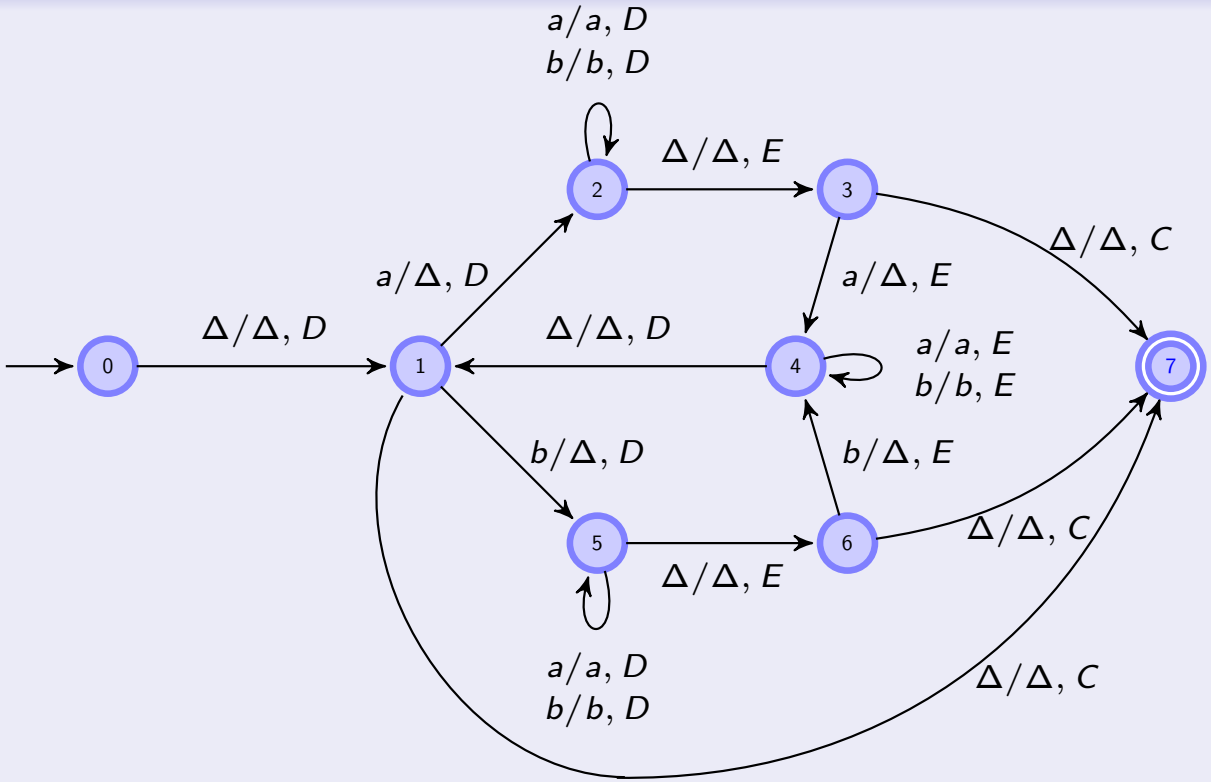
Toda a linguagem regular é reconhecida por uma máquina de Turing.

No entanto, nem todas as linguagens reconhecidas por máquinas de Turing são regulares. Tal é o caso, por exemplo, da linguagem reconhecida pela máquina de Turing do exemplo da página 13 (considerada também na página 18). Apresentaremos de seguida outra destas linguagens.

Exemplo

Seja $L = \{u \in A^* \mid u^I = u\}$ a linguagem não regular das palavras capicua sobre o alfabeto $A = \{a, b\}$. A seguinte máquina de Turing $\mathcal{T}$ reconhece a linguagem L

Exemplo (continuação)



A máquina $\mathcal{T}$ aceita a palavra *abba*, como o mostra a seguinte computação

$$(0, \underline{\Delta}abba) \longrightarrow (1, \Delta\underline{a}bba) \longrightarrow (2, \Delta\Delta\underline{b}ba) \overset{*}{\longrightarrow} (2, \Delta\Delta bba\underline{\Delta}) \longrightarrow (3, \Delta\Delta bb\underline{a}) \longrightarrow$$
$$(4, \Delta\Delta b\underline{b}\Delta) \overset{*}{\longrightarrow} (4, \Delta\underline{\Delta}bb\Delta) \longrightarrow (1, \Delta\Delta\underline{b}b\Delta) \longrightarrow (5, \Delta\Delta\Delta\underline{b}\Delta) \longrightarrow (5, \Delta\Delta\Delta b\underline{\Delta})$$
$$\longrightarrow (6, \Delta\Delta\Delta\underline{b}\Delta) \longrightarrow (4, \Delta\Delta\underline{\Delta}\Delta\Delta) \longrightarrow (1, \Delta\Delta\Delta\underline{\Delta}\Delta) \longrightarrow (7, \Delta\Delta\Delta\underline{\Delta}\Delta).$$

Analogamente, pode-se verificar que $\mathcal{T}$ aceita *aba* e não aceita *aaba*.

Exemplo (continuação)

Note-se que a estratégia desta máquina $\mathcal{T}$ para verificar se $u \in L$, para uma dada palavra $u \in A^*$, é a de iterar o seguinte procedimento:

- procurar a 1ª e a última letras da palavra em análise e verificar se são iguais;
- se o forem, estas letras são substituídas pelo símbolo branco e este procedimento repete-se.

A máquina parará por uma de três razões: em algum passo do processo,

1. as letras inicial e final da palavra em análise são diferentes. Neste caso $\mathcal{T}$ pára nos estados 3 ou 6 e a palavra u (que não é uma capicua) não é aceite.
2. a palavra em análise é a palavra vazia. Neste caso a máquina atinge o estado final (a partir do estado 1) e, portanto, a palavra u (que é uma capicua de comprimento par) é aceite.
3. a palavra em análise é uma letra. Neste caso a máquina atinge o estado final (a partir dos estados 3 ou 6) e, portanto, a palavra u (que é uma capicua de comprimento ímpar) é aceite.

Definição

Sejam $\mathcal{T}_1 = (Q_1, A, T, \delta_1, i_1, f_1, \Delta)$ e $\mathcal{T}_2 = (Q_2, A, T, \delta_2, i_2, f_2, \Delta)$ duas máquinas de Turing com $Q_1 \cap Q_2 = \emptyset$. A *composição sequencial* de $\mathcal{T}_1$ com $\mathcal{T}_2$ é a MT

$$\mathcal{T}_1\mathcal{T}_2 = (Q_1 \cup Q_2, A, T, \delta, i_1, f_2, \Delta)$$

cuja função transição δ é definida por

i) $\forall q \in Q_1 \ \forall t \in T$, se $\delta_1(q,t)$ está definido, então $\delta(q,t) = \delta_1(q,t)$;

ii) $\forall t \in T$, $\delta(f_1,t) = (i_2,t,C)$;

iii) $\forall q \in Q_2 \ \forall t \in T$, se $\delta_2(q,t)$ está definido, então $\delta(q,t) = \delta_2(q,t)$.

Informalmente, a máquina de Turing $\mathcal{T}_1\mathcal{T}_2$:

i) atua como a máquina $\mathcal{T}_1$ quando está posicionada num estado de $Q_1 \setminus \{f_1\}$;

ii) passa para i_2 sem alterar a situação da fita quando está posicionada em f_1;

iii) atua como a máquina $\mathcal{T}_2$ quando está posicionada num estado de Q_2.

Na representação de máquinas de Turing serão usadas as seguinte notações:

1. $\mathcal{T}_1 \longrightarrow \mathcal{T}_2$ para representar a composição sequencial de $\mathcal{T}_1$ com $\mathcal{T}_2$.
2. $\mathcal{T}_1 \xrightarrow{a/a,C} \mathcal{T}_2$ (onde $a \in T$) para representar a *composição sequencial condicional* de $\mathcal{T}_1$ com $\mathcal{T}_2$: isto é, a composição sequencial em que a condição *ii*) é substituída pela condição *ii'*) $\delta(f_1, a) = (i_2, a, C)$.
3. $(q) \longrightarrow \mathcal{T}_2$ e $(q) \xrightarrow{a/a,C} \mathcal{T}_2$ (onde $q \in Q_1$ e $a \in T$) para representar a máquina de Turing que se comporta como $\mathcal{T}_1$ até atingir o estado q e depois comporta-se como $\mathcal{T}_2$ (no segundo caso o cursor tem que ler um a).
4. $\mathcal{T}_1 \longrightarrow (q)$ e $\mathcal{T}_1 \xrightarrow{a/a,C} (q)$ (onde $q \in Q_2$ e $a \in T$) com um significado simétrico em relação ao do ponto anterior.

As máquinas de Turing que apresentamos de seguida são frequentemente utilizadas para construir novas MT, usando composição sequencial. As computações que realizam são úteis para fazerem parte de computações mais complexas.

Exemplo (Cópia de palavras)

Construamos uma máquina de Turing

$$\mathcal{T}_{\text{copiar}} = (Q, A, T, \delta, i, f, \Delta)$$

capaz de, ao partir da configuração inicial associada a uma palavra u, efetuar uma cópia de u na fita, colocando um símbolo branco entre o original e a cópia, ficando a cabeça posicionada na 1ª célula. Ou seja, $\mathcal{T}_{\text{copiar}}$ deverá ser capaz de efetuar a seguinte computação

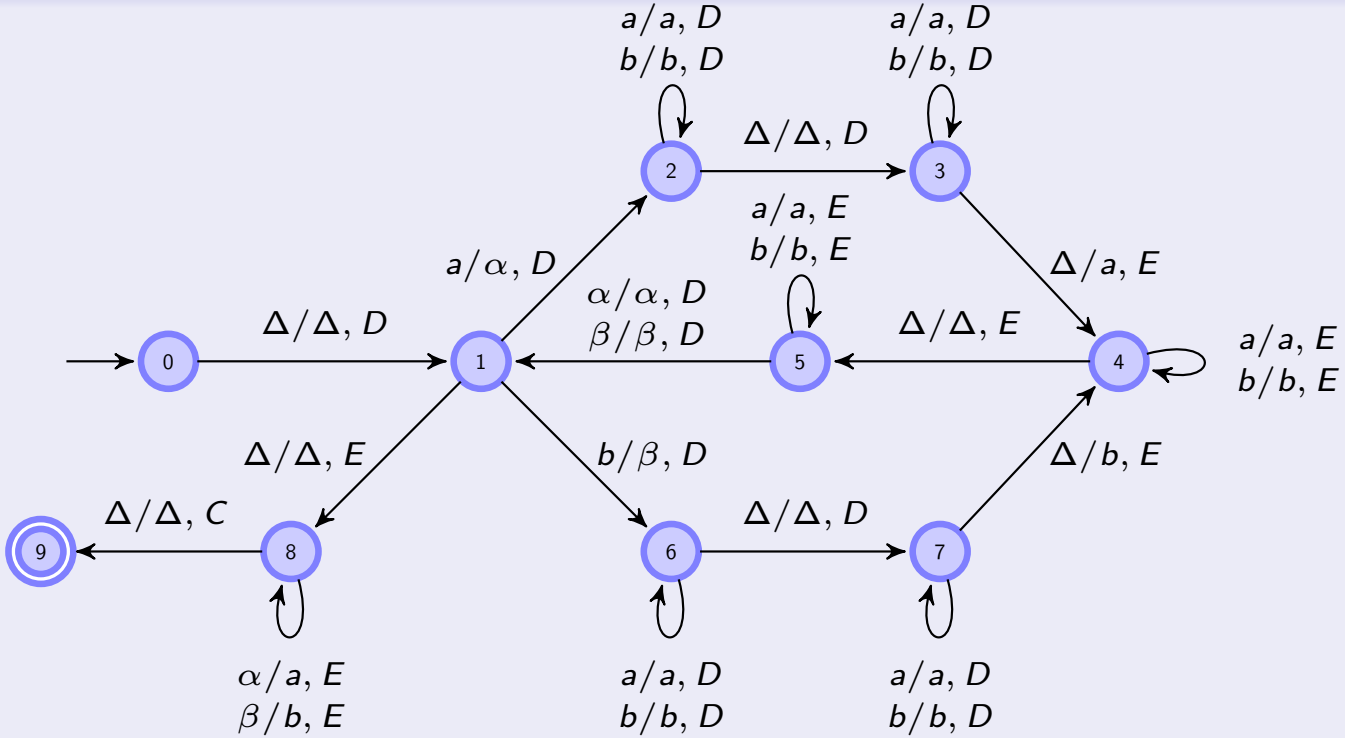
$$(i, \underline{\Delta} u) \xrightarrow{*} (f, \underline{\Delta} u \Delta u)$$

onde $u \in A^*$ é uma palavra qualquer sobre o alfabeto de entrada.

Admitamos que o alfabeto de entrada é $A = \{a, b\}$ (para alfabetos mais genéricos constroem-se máquinas de Turing análogas) e que α e β são letras do alfabeto auxiliar (que vão estar associadas a a e b respetivamente). A estratégia da máquina, indicada abaixo, é a de:

- copiar uma letra de u de cada vez,
- substituindo as letras já copiadas pelo correspondente símbolo auxiliar.

Exemplo (continuação)



Por exemplo, em $\mathcal{T}_{\text{copiar}}$ pode efetuar-se a seguinte computação

$$(0, \underline{\Delta}ba) \longrightarrow (1, \Delta\underline{b}a) \longrightarrow (6, \Delta\beta\underline{a}) \longrightarrow (6, \Delta\beta a\underline{\Delta}) \longrightarrow (7, \Delta\beta a\Delta\underline{\Delta}) \longrightarrow (4, \Delta\beta a\underline{\Delta}b)$$
$$\longrightarrow (5, \Delta\beta\underline{a}\Delta b) \longrightarrow (5, \Delta\underline{\beta}a\Delta b) \longrightarrow (1, \Delta\underline{\beta}a\Delta b) \longrightarrow (2, \Delta\beta\alpha\underline{\Delta}b) \longrightarrow (3, \Delta\beta\alpha\Delta\underline{b})$$
$$\longrightarrow (3, \Delta\beta\alpha\Delta b\underline{\Delta}) \longrightarrow (4, \Delta\beta\alpha\Delta\underline{b}a) \longrightarrow (4, \Delta\beta\alpha\underline{\Delta}ba) \longrightarrow (5, \Delta\beta\underline{\alpha}\Delta ba) \longrightarrow$$
$$(1, \Delta\beta\alpha\underline{\Delta}ba) \longrightarrow (8, \Delta\beta\underline{\alpha}\Delta ba) \longrightarrow (8, \Delta\underline{\beta}a\Delta ba) \longrightarrow (8, \underline{\Delta}ba\Delta ba) \longrightarrow (9, \underline{\Delta}ba\Delta ba).$$

Exemplo (Apagar letras)

Construamos uma máquina de Turing

$$\mathcal{T}_{\text{apagar}} = (Q, A, T, \delta, i, f, \Delta)$$

que apague a letra que o cursor está a ler, ou seja, que faça a computação

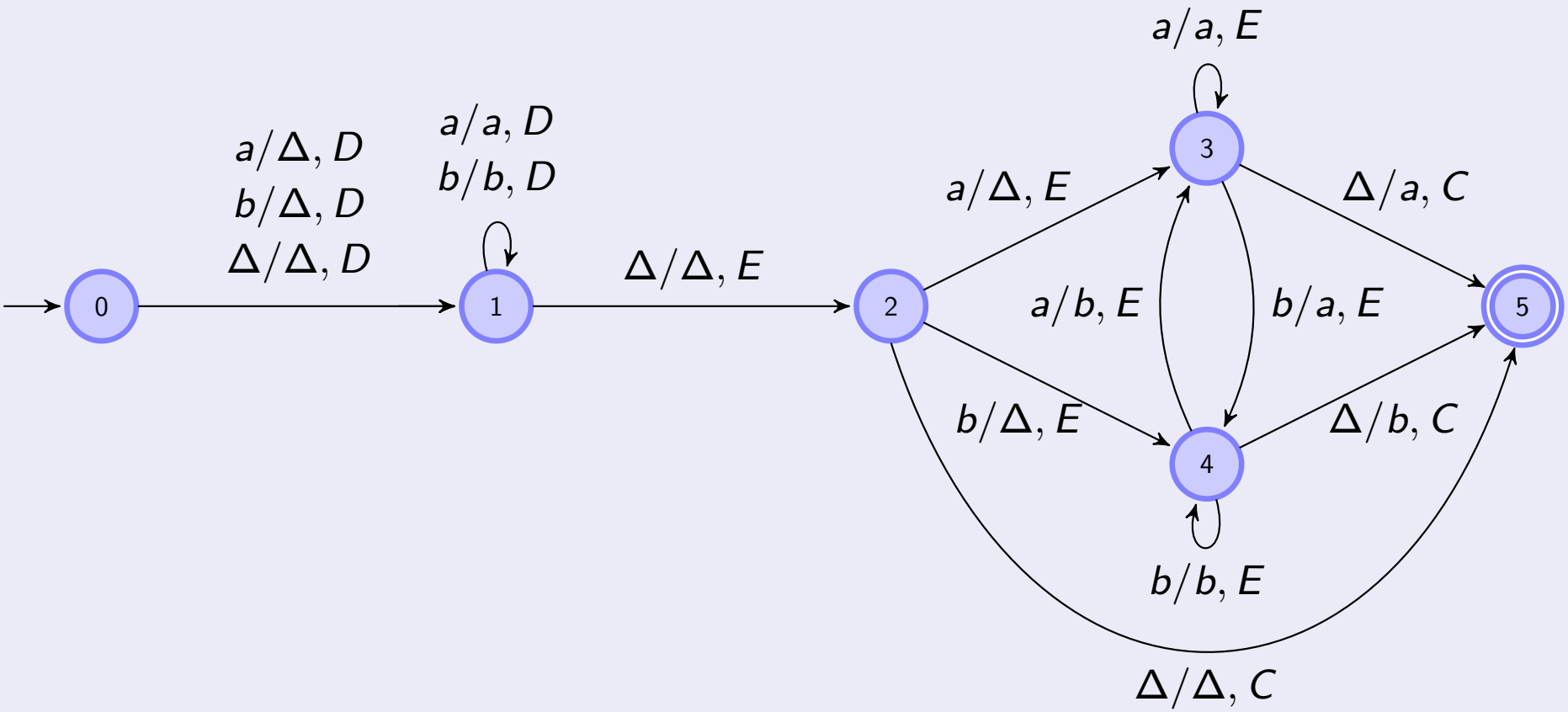
$$(i, u\underline{t}v) \xrightarrow{*} (f, u\underline{v})$$

em que $u \in (A \cup \{\Delta\})^*$, $t \in A \cup \{\Delta\}$ e $v \in A^*$.

Admitamos que $A = \{a, b\}$ e que o único símbolo auxiliar é Δ. A máquina:

- começa por substituir a letra t a apagar pelo símbolo branco,
- desloca o cursor para a última letra de v e
- move sucessivamente cada letra de v uma célula para a esquerda.

Exemplo (continuação)



Exercício

Calcule a sequência de computações em $\mathcal{T}_{\text{apagar}}$ a partir da configuração $(0, \Delta b\underline{a}bbaab)$.

Definição

Seja $\mathcal{T} = (Q, A, T, \delta, i, f, \Delta)$ uma máquina de Turing e seja

$$g : A^* \to T^*$$

uma função parcial. Diz-se que $\mathcal{T}$ *calcula* g quando, para cada $u \in A^*$:

- se g está definida para u, então pode ser efetuada a computação

$$(i, \underline{\Delta} u) \xrightarrow[\mathcal{T}]{*} (f, \underline{\Delta} g(u)).$$

- se g não está definida para u, então não é possível computar uma configuração de aceitação a partir da configuração inicial associada a u.

Definição(continuação)

Mais geralmente, para um natural k e uma função parcial em k variáveis

$$g : (A^*)^k \to T^*,$$

diz-se que $\mathcal{T}$ *calcula* g quando, para cada $u_1, \ldots, u_k \in A^*$:

- se $g(u_1, \ldots, u_k)$ está definida, então pode ser efetuada a computação
$$(i, \underline{\Delta} u_1 \Delta \cdots \Delta u_k) \xrightarrow[\mathcal{T}]{*} (f, \underline{\Delta} g(u_1, \ldots, u_k)).$$
- se $g(u_1, \ldots, u_k)$ não está definida, então não é possível computar uma configuração de aceitação a partir da configuração $(i, \underline{\Delta} u_1 \Delta \cdots \Delta u_k)$.

Definição

Sejam $k \in \mathbb{N}$, A e T alfabetos com $A \subseteq T$ e $g : (A^*)^k \to T^*$ 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.

Exemplo (Concatenação de palavras)

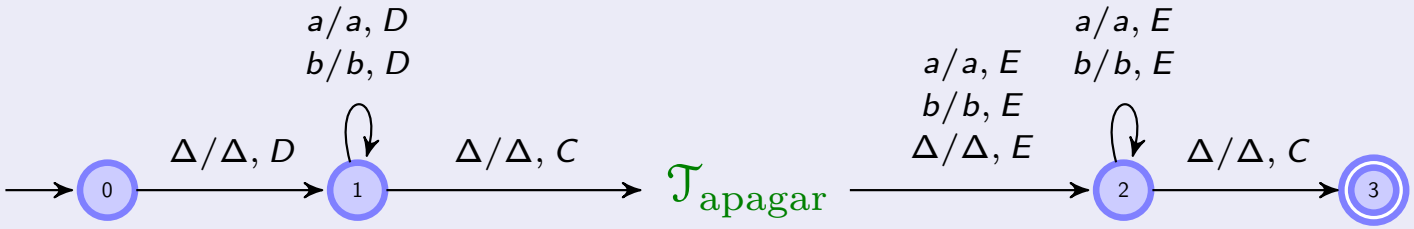
Seja $A = \{a, b\}$ e seja

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

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

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

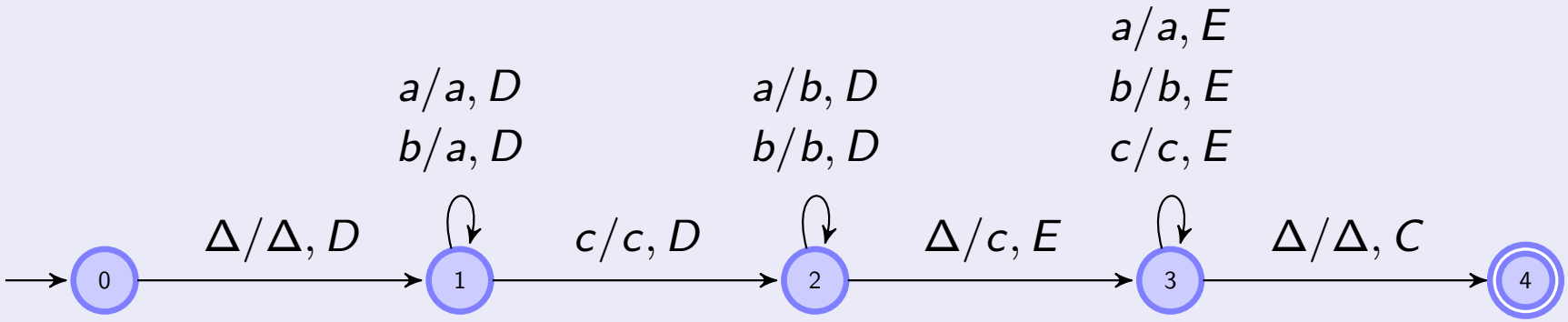
para cada $u, v \in A^*$. A ideia é que a máquina $\mathcal{T}_{\text{conc}}$, apresentada abaixo,



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*.

Exercício

Seja $\mathcal{T}$ a seguinte máquina de Turing sobre o alfabeto $A = \{a, b, c\}$,



A máquina $\mathcal{T}$ calcula uma função $g : A^* \to A^*$.

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

b) Identifique o domínio D da função g.

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

Resposta (sucinta)

a) $(0, \underline{\Delta}abacbbaba) \xrightarrow{*} (4, \underline{\Delta}a^3cb^5c)$.

b) $D = \{a, b\}^* c \{a, b\}^*$.

c) Se $u \in D$, então $u = vcw$ com $v, w \in \{a, b\}^*$. Logo $g(u) = a^{|v|}cb^{|w|}c$.

- 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

$$n \in \mathbb{N}_0 \text{ é representado pela palavra } 1^n = \underbrace{11\cdots1}_{n \text{ vezes}}.$$

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

Definição

Seja $\mathcal{T} = (Q, \{1\}, T, \delta, i, f, \Delta)$ uma máquina de Turing e seja $g : \mathbb{N}_0^k \to \mathbb{N}_0$ uma função parcial. Diz-se que $\mathcal{T}$ *calcula* g quando, para $n_1, \ldots, n_k \in \mathbb{N}_0$:

- se $g(n_1, \ldots, n_k)$ está definida, então pode ser efetuada a computação

$$(i, \underline{\Delta}1^{n_1}\Delta\cdots\Delta1^{n_k}) \xrightarrow[\mathcal{T}]{*} (f, \underline{\Delta}1^{g(n_1,\ldots,n_k)}).$$

- se $g(n_1, \ldots, n_k)$ não está definida, então não é possível computar uma configuração de aceitação a partir da configuração $(i, \underline{\Delta}1^{n_1}\Delta\cdots\Delta1^{n_k})$.

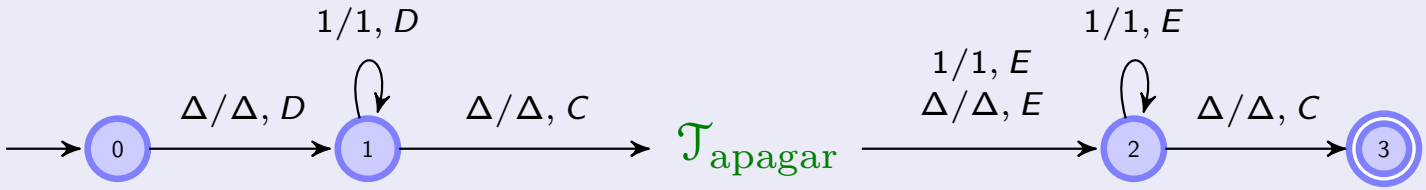
Exemplo (Adição em $\mathbb{N}_0$)

- Consideremos a função

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

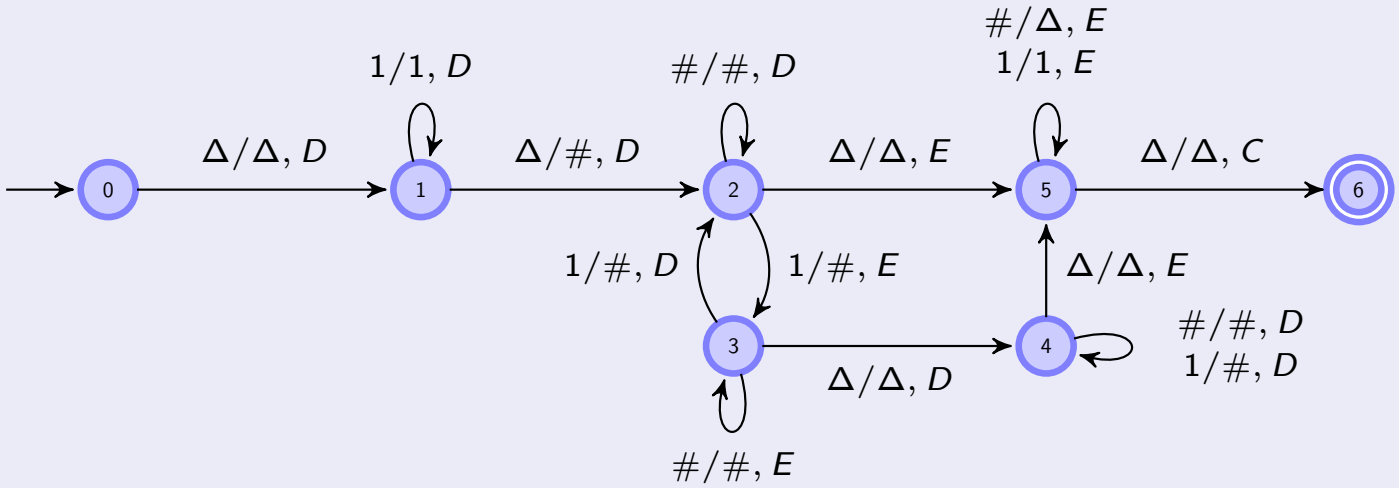
de adição em $\mathbb{N}_0$.

- A seguinte máquina de Turing calcula $+$



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

Exercício



A máquina de Turing acima calcula uma função $g : \mathbb{N}_0^2 \to \mathbb{N}_0$.

a) Indique as computações que podem ser feitas a partir das configurações:
i) $(0, \underline{\Delta}111\Delta1)$; *ii*) $(0, \underline{\Delta}1\Delta111)$.

b) Identifique o domínio D da função g.

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

Resposta (sucinta)

a)*i*) $(0, \underline{\Delta}111\Delta1) \xrightarrow{*} (6, \underline{\Delta}11)$; *ii*) $(0, \underline{\Delta}1\Delta111) \xrightarrow{*} (6, \underline{\Delta})$.

b)*c*) $D = \mathbb{N}_0^2$, $g(x, y) = monus(x, y) = x \dot{-} y = \begin{cases} x - y & \text{se } x \geq y \\ 0 & \text{se } x < y \end{cases}$.

Definição

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

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

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

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

Exemplo

Consideremos a seguinte linguagem L sobre o alfabeto $A = \{a, b, c\}$,

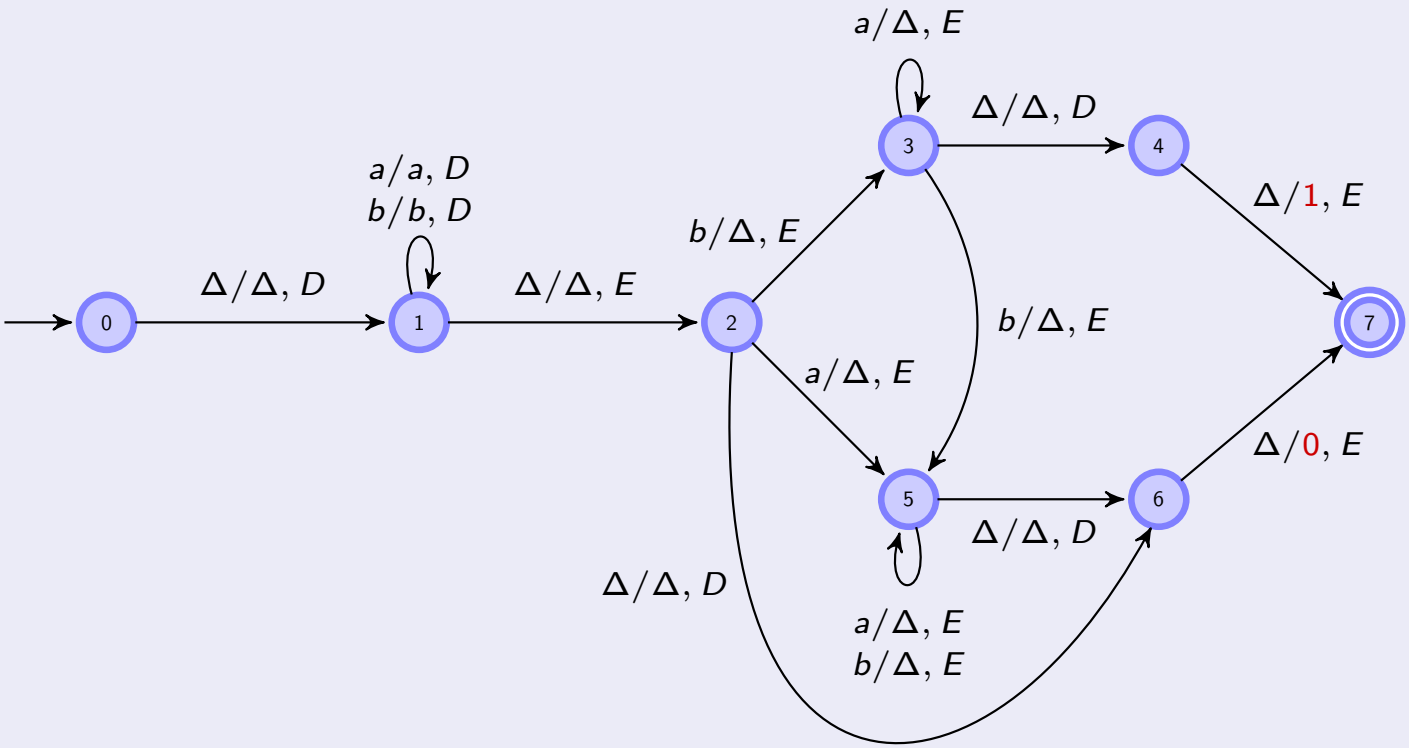
$$L = \{wa^n : w \in A^*, n \in \mathbb{N}_0, |w|_b = n\}.$$

Tem-se, por exemplo,

$$\chi_L(aacac) = 1, \quad \chi_L(abbcbaa) = 0, \quad \chi_L(abcbaaaaa) = 1,$$
$$\chi_L(abbcaacb) = 0, \quad \chi_L(baacabaa) = 1, \quad \chi_L(bcbaca) = 0.$$

Exemplo (Cálculo da função característica)

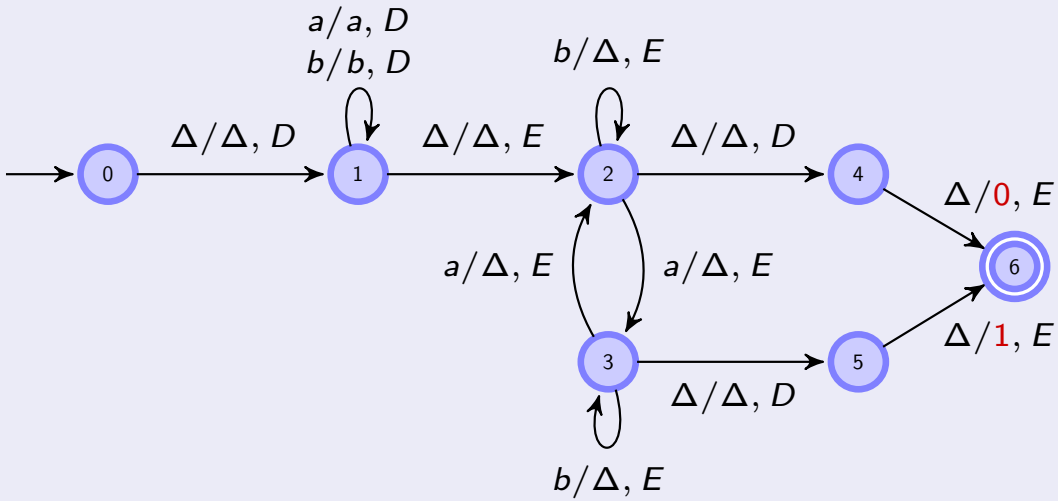
Sejam $A = \{a, b\}$ e $L = a^*b$. A seguinte máquina de Turing calcula χ_L.



Note-se que $a^3b \in L$ e $ab^2a^3b \notin L$. Na MT podem efetuar-se as computações:

- $(0, \underline{\Delta}a^3b) \longrightarrow (1, \Delta\underline{a}a^2b) \xrightarrow{*} (1, \Delta a^3b\underline{\Delta}) \longrightarrow (2, \Delta a^3\underline{b}) \longrightarrow (3, \Delta a^2\underline{a}) \xrightarrow{*} (3, \underline{\Delta}) \longrightarrow (4, \Delta\underline{\Delta}) \longrightarrow (7, \underline{\Delta}1)$;
- $(0, \underline{\Delta}ab^2a^3b) \longrightarrow (1, \Delta\underline{a}b^2a^3b) \xrightarrow{*} (1, \Delta ab^2a^3b\underline{\Delta}) \longrightarrow (2, \Delta ab^2a^3\underline{b}) \longrightarrow (3, \Delta ab^2a^2\underline{a}) \xrightarrow{*} (3, \Delta ab\underline{b}) \longrightarrow (5, \Delta a\underline{b}) \xrightarrow{*} (5, \underline{\Delta}) \longrightarrow (6, \Delta\underline{\Delta}) \longrightarrow (7, \underline{\Delta}0)$.

Exercício



A MT acima calcula a função característica χ_L de uma linguagem $L \subseteq \{a, b\}^*$.

a) Indique as configurações que podem ser computadas a partir de $(0, \underline{\Delta} aba^2 b)$.

b) Determine o valor de $\chi_L(bab^3aba^2b^5)$.

c) Identifique a linguagem L.

Resposta (sucinta)

a) $(0, \underline{\Delta} aba^2 b) \xrightarrow{*} (2, \Delta aba^2 \underline{b}) \xrightarrow{*} (3, \underline{\Delta}) \longrightarrow (5, \Delta\underline{\Delta}) \longrightarrow (6, \underline{\Delta}1)$.

b) $\chi_L(bab^3aba^2b^5) = 0$.

c) $L = \{w \in \{a, b\}^* : |w|_a \text{ é ímpar}\}$.

- A afirmação abaixo foi enunciada pela primeira vez por Alonzo Church na década de 1930 e é conhecida como *tese de Church* ou *tese de Church-Turing*.
- Embora não seja um enunciado matemático preciso e, por isso, não possa ser provada, é geralmente aceite como válida pois não há qualquer evidência de que esteja errada.

Tese de Church

Qualquer função que pode ser calculada por um processo algorítmico, por pessoas ou computadores, é Turing-computável.

- O modelo de máquina de Turing que temos vindo a estudar admite várias alternativas possíveis.
- Consideraremos de seguida duas dessas alternativas.

Definição

- Uma *máquina de Turing com ℓ fitas* é um septeto $\mathcal{T} = (Q, A, T, \delta, i, f, \Delta)$ similar a uma MT usual, mas com ℓ fitas independentes, cada uma com o seu cursor, em que a função transição δ é uma função parcial

$$\delta : Q \times T^{\ell} \to Q \times T^{\ell} \times \{E, C, D\}^{\ell}.$$

- Uma *configuração* de $\mathcal{T}$ é um $(\ell + 1)$-tuplo ordenado

$$(q, u_1 \underline{v_1}, \ldots, u_{\ell} \underline{v_{\ell}})$$

onde $q \in Q$ e $u_i, v_i \in T^*$, em que:

  - q é o estado atual de $\mathcal{T}$;
  - para cada $i \in \{1, \ldots, \ell\}$ a $(i+1)$-ésima componente representa a situação da *i-ésima fita* e do cursor correspondente.

- As noções de *computação direta*, *computação*, *configuração de paragem* (resp. *de aceitação*, *de rejeição*, *de ciclo*) são similares às noções correspondentes para máquinas de Turing usuais.

- A *configuração inicial* associada a uma palavra $u \in A^*$ é

$$(i, \underline{\Delta} u, \underline{\Delta}, \ldots, \underline{\Delta}).$$

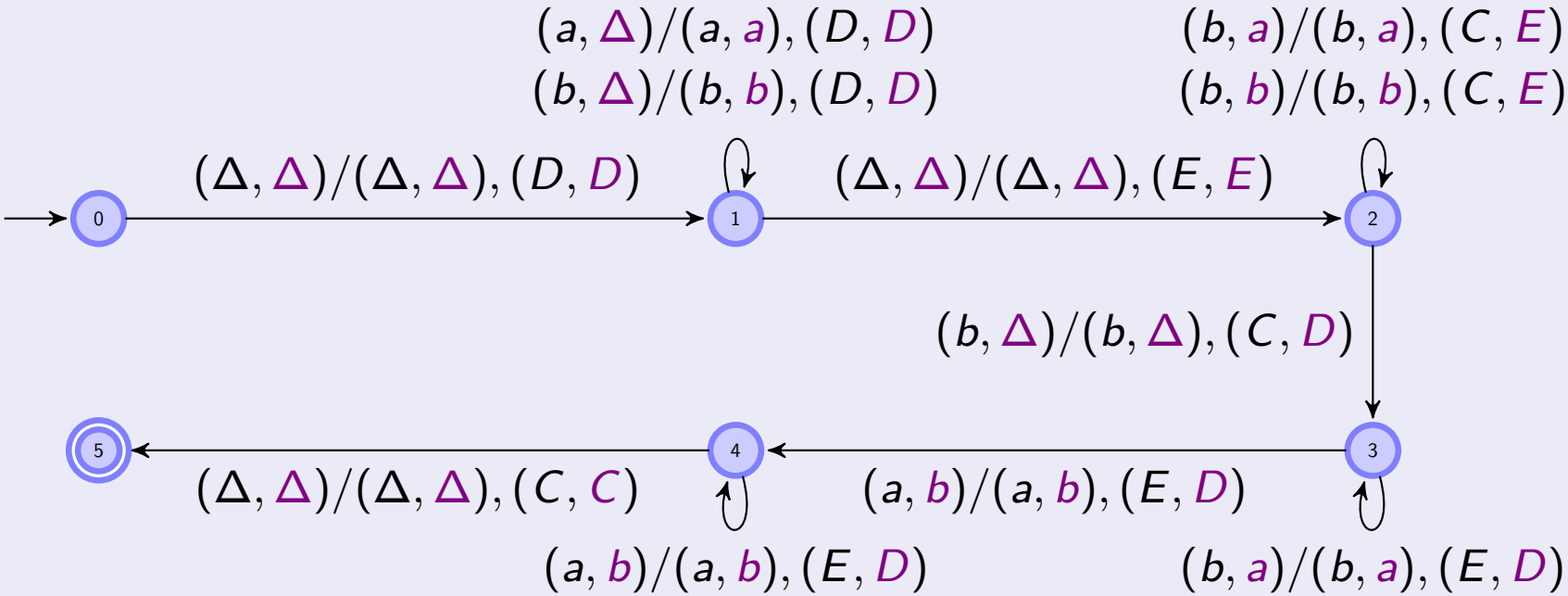
- Uma palavra $u \in A^*$ é *aceite* por $\mathcal{T}$ se

$$(i, \underline{\Delta} u, \underline{\Delta}, \ldots, \underline{\Delta}) \xrightarrow[\mathcal{T}]{*} (f, v_1 \underline{w_1}, \ldots, v_\ell \underline{w_\ell})$$

  para alguns $v_1, \ldots, v_\ell, w_1, \ldots, w_\ell \in T^*$, ou seja, se existe uma computação de uma configuração de aceitação a partir da configuração inicial de u.

- A linguagem *aceite* ou *reconhecida* por $\mathcal{T}$, é o conjunto das palavras $L \subseteq A^*$ formado pelas palavras aceites por $\mathcal{T}$.

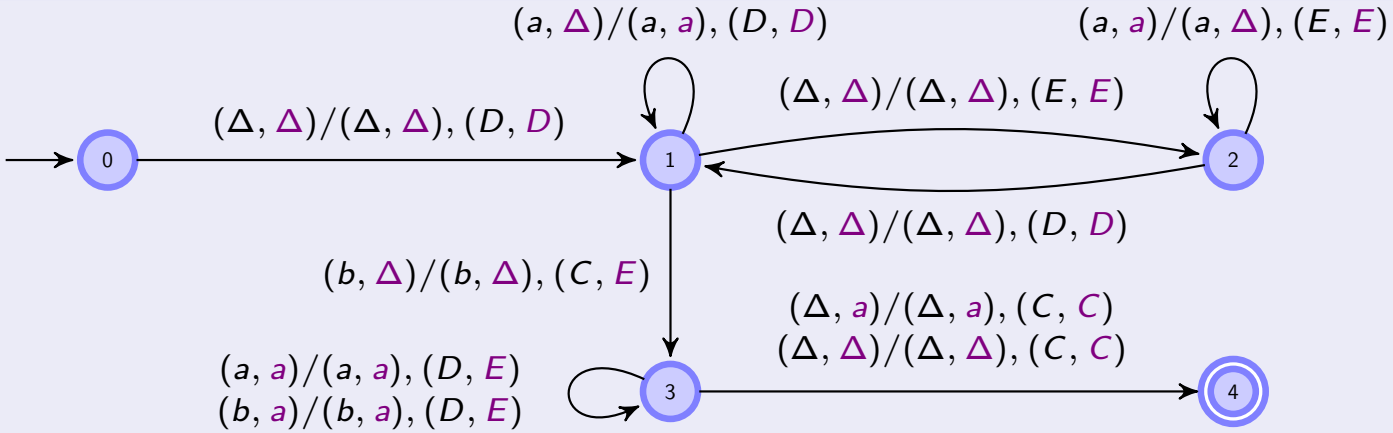
Exemplo



A figura acima apresenta uma MT $\mathcal{T}$ sobre $\{a, b\}$ com 2 fitas, que aceita a linguagem $\{a^n b^n \mid n \in \mathbb{N}\}$. Por exemplo, em $\mathcal{T}$ pode efetuar-se a computação

$$(0, \underline{\Delta}aabb, \underline{\Delta}) \longrightarrow (1, \Delta\underline{a}abb, \Delta\underline{\Delta}) \xrightarrow{*} (1, \Delta aabb\underline{\Delta}, \Delta aabb\underline{\Delta}) \longrightarrow$$
$$(2, \Delta aab\underline{b}, \Delta aab\underline{b}) \xrightarrow{*} (2, \Delta aab\underline{b}, \underline{\Delta}aabb) \longrightarrow (3, \Delta aab\underline{b}, \Delta\underline{a}abb) \longrightarrow$$
$$(3, \Delta aa\underline{b}b, \Delta a\underline{a}bb) \longrightarrow (3, \Delta a\underline{a}bb, \Delta aa\underline{b}b) \longrightarrow (4, \Delta\underline{a}abb, \Delta aab\underline{b}) \longrightarrow$$
$$(4, \underline{\Delta}aabb, \Delta aabb\underline{\Delta}) \longrightarrow (5, \underline{\Delta}aabb, \Delta aabb\underline{\Delta}).$$

Exercício



Seja $\mathcal{T}$ a MT acima com duas fitas sobre $A = \{a, b\}$.

a) Indique as configurações que podem ser computadas a partir de $(0, \underline{\Delta}aaaabab, \underline{\Delta})$ e diga se a palavra $aaaabab$ é aceite por $\mathcal{T}$.

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

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

d) Identifique a linguagem L reconhecida por $\mathcal{T}$.

Resposta sucinta

a) $(0, \underline{\Delta}a^4bab, \underline{\Delta}) \xrightarrow{*} (4, \Delta a^4bab\underline{\Delta}, \Delta\underline{a}a^3)$; $aaaabab$ é aceite; *b)* $u \in a^*$;

c) $v \in \{a^n bw : n \in \mathbb{N}_0, w \in A^*, n < |bw|\}$; *d)* $L = \{a^n bw : n \in \mathbb{N}, w \in A^*, n \geq |bw|\}$.

Definição

Sendo $k \in \mathbb{N}$ e $g : (A^*)^k \to T^*$ uma função parcial em k variáveis, diz-se que $\mathcal{T}$ *calcula* g quando, para cada $u_1, \ldots, u_k \in A^*$:

- se $g(u_1, \ldots, u_k)$ está definida, então pode ser efetuada a computação

$$(i, \underline{\Delta} u_1 \Delta u_2 \Delta \cdots \Delta u_k, \underline{\Delta}, \ldots, \underline{\Delta}) \xrightarrow[\mathcal{T}]{*} (f, \underline{\Delta}\ g(u_1, \ldots, u_k), v_2 \underline{w_2}, \ldots, v_\ell \underline{w_\ell})$$

  para alguns $v_2, \ldots, v_\ell, w_2, \ldots, w_\ell \in T^*$.

- se $g(u_1, \ldots, u_k)$ não está definida, então não é possível computar uma configuração de aceitação a partir da configuração $(i, \underline{\Delta} u_1 \Delta \cdots \Delta u_k, \underline{\Delta}, \ldots, \underline{\Delta})$.

Teorema

As máquinas de Turing usuais e as máquinas de Turing com ℓ fitas reconhecem as mesmas linguagens e calculam as mesmas funções.

Definição

Uma *máquina de Turing não-determinista* é um septeto

$$\mathcal{T} = (Q, A, T, \delta, i, f, \Delta)$$

definido de forma similar a uma máquina de Turing usual com exceção da função transição δ que é da forma

$$\delta : Q \times T \to \mathcal{P}(Q \times T \times \{E, C, D\}).$$

As noções de *computação*, de *aceitação de palavras* e *linguagens*, e de *cálculo de funções* são similares às correspondentes noções para as máquinas de Turing usuais.

A diferença entre as máquinas de Turing não-deterministas e as máquinas de Turing usuais é que a partir de uma dada configuração:

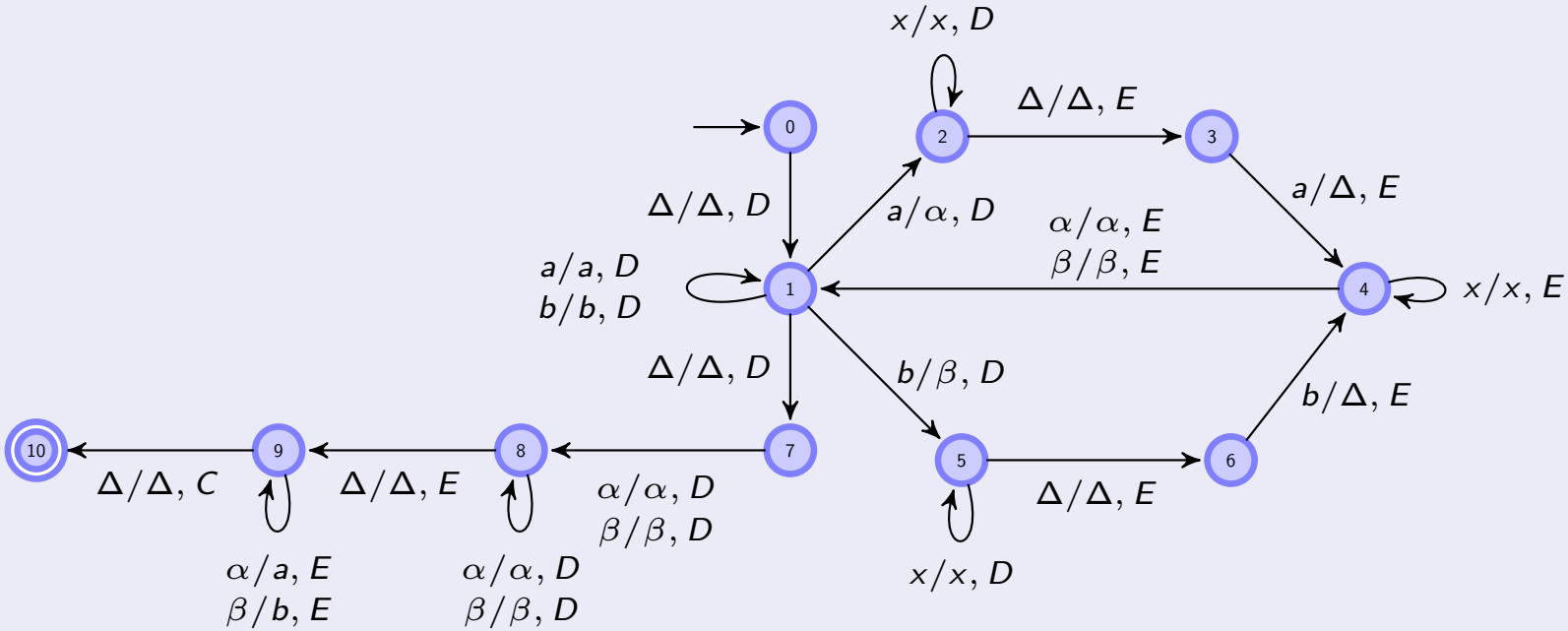
- nas máquinas de Turing usuais só pode ser efetuada, no máximo, uma computação direta;
- nas máquinas de Turing não-deterministas podem ser efetuadas, eventualmente, várias computações diretas.

Teorema

As máquinas de Turing usuais e as máquinas de Turing não-deterministas reconhecem as mesmas linguagens e calculam as mesmas funções.

Exemplo

Seja $\mathcal{T}$ a seguinte MT não-determinista sobre $A = \{a, b\}$, onde $x \in \{a, b, \alpha, \beta\}$,

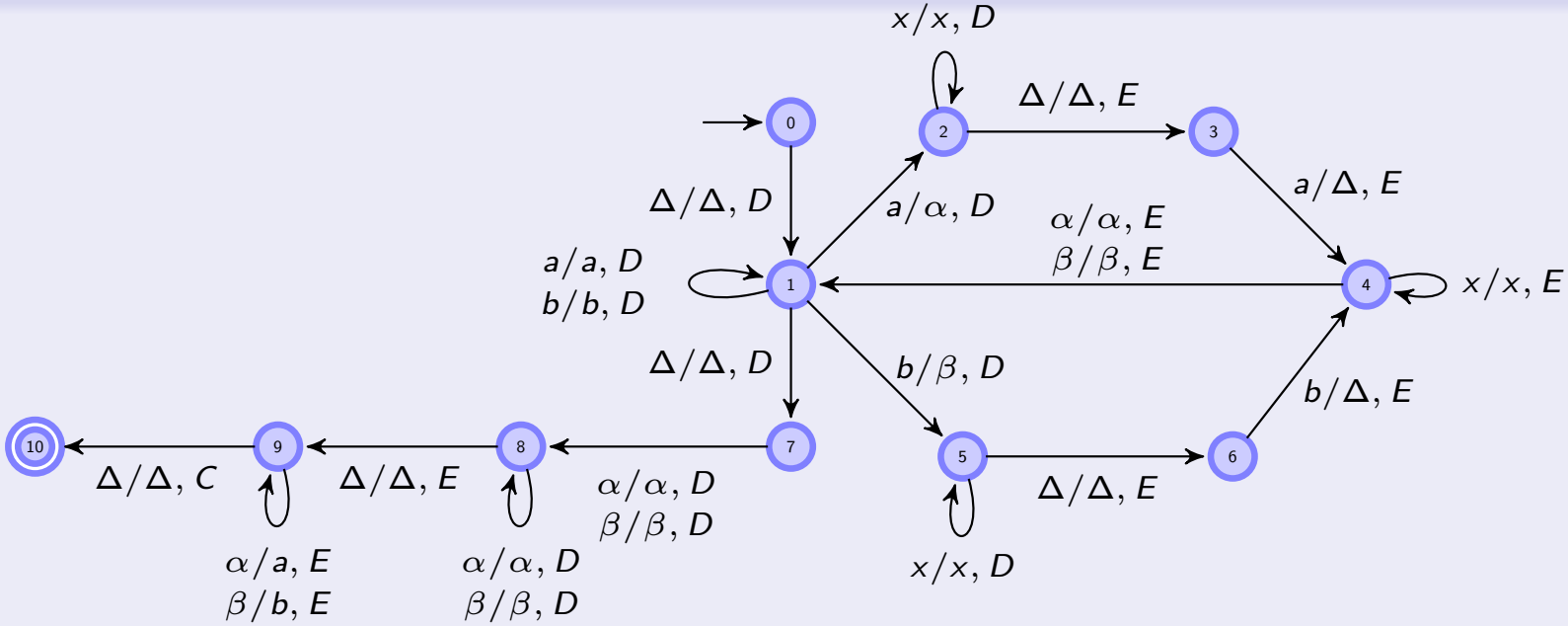


Em $\mathcal{T}$ pode efetuar-se, por exemplo, a computação

$$(0, \underline{\Delta}abbabb) \longrightarrow (1, \Delta\underline{a}bbabb) \longrightarrow (1, \Delta a\underline{b}babb) \longrightarrow (5, \Delta a\beta\underline{b}abb) \xrightarrow{*}$$
$$(5, \Delta a\beta babb\underline{\Delta}) \longrightarrow (6, \Delta a\beta bab\underline{b}) \longrightarrow (4, \Delta a\beta ba\underline{b}) \xrightarrow{*} (4, \Delta a\underline{\beta}bab) \longrightarrow$$
$$(1, \Delta\underline{a}\beta bab) \longrightarrow (2, \Delta\alpha\underline{\beta}bab) \xrightarrow{*} (2, \Delta\alpha\beta bab\underline{\Delta}) \longrightarrow (3, \Delta\alpha\beta ba\underline{b}).$$

Dado que $(3, \Delta\alpha\beta ba\underline{b})$ é uma configuração de rejeição, quererá isto significar que a palavra $abbabb$ não é aceite pela máquina $\mathcal{T}$? Não necessariamente!...

Exemplo (continuação)



De facto, em $\mathcal{T}$ pode também efetuar-se a computação

$$(0, \underline{\Delta}abbabb) \longrightarrow (1, \Delta\underline{a}bbabb) \xrightarrow{*} (1, \Delta ab\underline{b}abb) \longrightarrow (5, \Delta ab\beta\underline{a}bb) \xrightarrow{*} (5, \Delta ab\beta abb\underline{\Delta}) \longrightarrow (6, \Delta ab\beta ab\underline{b}) \longrightarrow$$
$$(4, \Delta ab\beta a\underline{b}) \xrightarrow{*} (4, \Delta ab\underline{\beta}ab) \longrightarrow (1, \Delta a\underline{b}\beta ab) \longrightarrow (5, \Delta a\beta\underline{\beta}ab) \xrightarrow{*} (5, \Delta a\beta\beta ab\underline{\Delta}) \longrightarrow (6, \Delta a\beta\beta a\underline{b}) \longrightarrow$$
$$(4, \Delta ab\beta\underline{a}) \xrightarrow{*} (4, \Delta a\underline{\beta}\beta a) \longrightarrow (1, \Delta\underline{a}\beta\beta a) \longrightarrow (2, \Delta\alpha\underline{\beta}\beta a) \xrightarrow{*} (2, \Delta\alpha\beta\beta a\underline{\Delta}) \longrightarrow (3, \Delta\alpha\beta\beta\underline{a}) \longrightarrow (4, \Delta\alpha\beta\underline{\beta})$$
$$\xrightarrow{*} (4, \Delta\underline{\alpha}\beta\beta) \longrightarrow (1, \underline{\Delta}\alpha\beta\beta) \longrightarrow (7, \Delta\underline{\alpha}\beta\beta) \xrightarrow{*} (8, \Delta\alpha\beta\beta\underline{\Delta}) \longrightarrow (9, \Delta\alpha\beta\underline{\beta}) \longrightarrow (9, \underline{\Delta}abb) \longrightarrow (10, \underline{\Delta}abb).$$

Como $(10, \underline{\Delta}abb)$ é uma configuração de aceitação, conclui-se que $abbabb$ é aceite por $\mathcal{T}$. Pode ainda mostrar-se que $\mathcal{T}$:

- reconhece a linguagem $L = \{uu : u \in A^+\}$;
- calcula a função parcial $g : A^* \to A^*$ definida, para cada $uu \in L$, por $g(uu) = u$.

Definição

Seja $L \subseteq A^*$ uma linguagem e seja $\mathcal{T}$ uma máquina de Turing com alfabeto de entrada A. Diz-se que:

- $\mathcal{T}$ *aceita* ou *reconhece* L se $L = L(\mathcal{T})$;
- $\mathcal{T}$ *decide* L se a função característica χ_L é calculada por $\mathcal{T}$.

Definição

Uma linguagem L diz-se:

- *recursivamente enumerável* se existe uma MT que reconhece L;
- *recursiva* (ou *decidível*) se existe uma MT que decide L.

Por exemplo, as linguagens regulares a^*b e $\{w \in A^* : |w|_a \text{ é ímpar}\}$ sobre o alfabeto $A = \{a, b\}$, consideradas nas páginas 40 e 41, são *recursivas*.

Proposição

Todas as linguagens regulares são recursivas.

Proposição

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 $u \in \overline{L}$.

Demonstração: ($\Rightarrow$) Se L é recursiva, então existe uma MT que decide L. Esta MT é um algoritmo e é fácil transformá-la num algoritmo que reconhece L.

($\Leftarrow$) A partir da configuração inicial associada a uma palavra $u \in A^*$ atinge-se sempre uma configuração de paragem. Esta configuração é de aceitação se $u \in L$ e é de rejeição se $u \notin L$. Modifica-se então a máquina de Turing $\mathcal{T}$ de forma que:

- a partir de uma configuração de aceitação em $\mathcal{T}$ se escreva 1 na 2ª célula da fita, deixando todas as outras células em branco e o cursor na 1ª célula;
- a partir de uma configuração de rejeição em $\mathcal{T}$ se escreva 0 na 2ª célula da fita, deixando todas as outras células em branco e o cursor na 1ª célula;

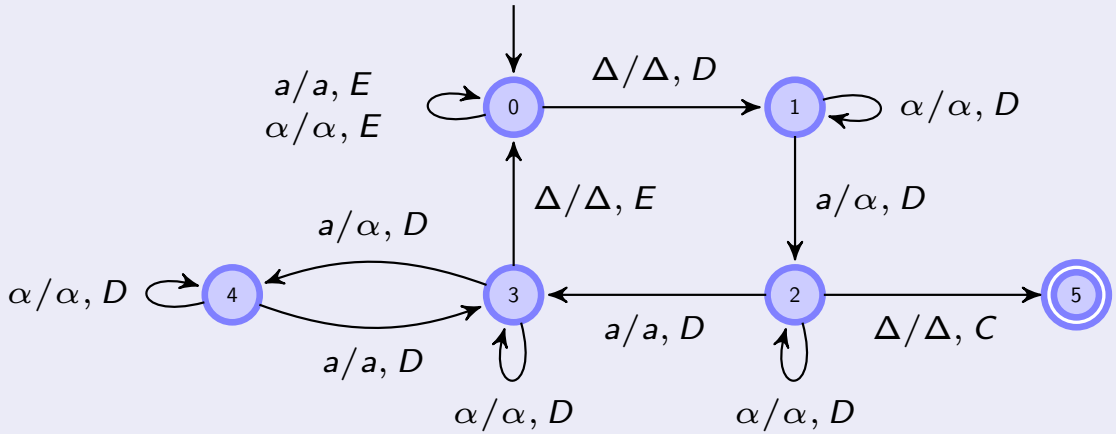
Esta nova MT calcula χ_L, a função característica de L, donde L é recursiva. □

Corolário

Todas as linguagens recursivas são recursivamente enumeráveis.

Exercício

Seja $\mathcal{T}$ a seguinte MT sobre o alfabeto $A = \{a\}$,



a) Indique as configurações que podem ser computadas a partir de $(0, \underline{\Delta} a^8)$.

b) Identifique a linguagem L reconhecida pela máquina $\mathcal{T}$.

c) A linguagem L é recursiva? Porquê?

Resposta (sucinta)

a) $(0, \underline{\Delta} a^8) \xrightarrow{*} (3, \Delta\alpha a\alpha a\alpha a\alpha a\underline{\Delta}) \xrightarrow{*} (0, \underline{\Delta}\alpha a\alpha a\alpha a\alpha a) \xrightarrow{*} (3, \Delta\alpha\alpha\alpha a\alpha\alpha\alpha a\underline{\Delta}) \xrightarrow{*} (0, \underline{\Delta}\alpha\alpha\alpha a\alpha\alpha\alpha a)$
$\xrightarrow{*} (3, \Delta\alpha\alpha\alpha\alpha\alpha\alpha\alpha a\underline{\Delta}) \xrightarrow{*} (0, \underline{\Delta}\alpha\alpha\alpha\alpha\alpha\alpha\alpha a) \xrightarrow{*} (2, \Delta\alpha^8\underline{\Delta}) \longrightarrow (5, \Delta\alpha^8\underline{\Delta})$.

b) $L = \{a^{2^n} : n \in \mathbb{N}_0\}$.

c) L é recursiva pois é aceite por $\mathcal{T}$ que é um algoritmo por nunca entrar em ciclo.

Proposição

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 $\overline{L}$ 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 $\mathcal{T}_K$.

A alínea *ii*) é imediata. □

Teorema [Post, 1943]

Uma linguagem L é recursiva se e só se L e $\overline{L}$ 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.

Teorema

Existem linguagens não recursivamente enumeráveis sobre qualquer alfabeto A. Mais precisamente, o conjunto das linguagens não recursivamente enumeráveis sobre A tem o cardinal do contínuo.

Demonstração: Basta notar que o conjunto $RE(A)$ é numerável, enquanto que $\mathcal{P}(A^*)$, o conjunto de todas as linguagens sobre A, tem o cardinal do contínuo. □

Convenção

Assume-se que existem dois conjuntos numeráveis

$$\mathcal{Q} = \{q_0, q_1, q_2, \ldots\} \quad \text{e} \quad \mathcal{S} = \{s_0, s_1, s_2, \ldots\}$$

tais que, para cada máquina de Turing,

$$\mathcal{T} = (Q, A, T, \delta, i, f, \Delta)$$

se tem:

- $Q \subseteq \mathcal{Q}$, com $f = q_0$;
- $T \subseteq \mathcal{S}$, com $\Delta = s_0$.

Diz-se que $\mathcal{T}$ é *normalizada* se todos os estados e todos os símbolos não brancos de $\mathcal{T}$ pertencem a alguma transição.

Observação

Cada máquina de Turing é equivalente a alguma máquina de Turing normalizada.

Definição [função codificadora]

Define-se uma função injetiva

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

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 $c'(\_)$, sobre o alfabeto $\{x\}$,

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

$$c'(q_i) = c'(s_i) = x^{i+1},$$
$$c'(C) = x,\ c'(E) = x^2,\ c'(D) = x^3.$$

  Note-se que, em particular,

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

- a cada transição e, descrita por $\delta(q,t) = (q',t',m)$:

$$c'(e) = c'(q)yc'(t)yc'(q')yc'(t')yc'(m)y$$

Definição (continuação)

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

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

onde q_i é o estado inicial de $\mathcal{T}$ e $e_1, e_2, \ldots, e_k$ 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 $r_i \in \mathcal{S}$, por

$$\boxed{c(w) = yyc'(r_1)yc'(r_2)\cdots yc'(r_n)y.}$$

Observação

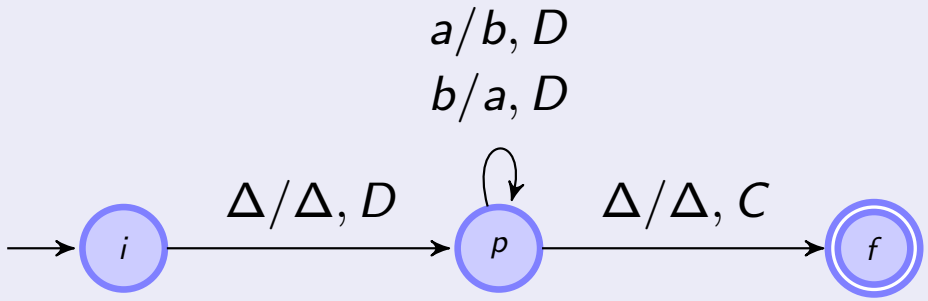
Quando se considera uma sequência

$$\boxed{c(\mathcal{T})c(w) = c'(q_i)yc'(e_1)yc'(e_2)\cdots yc'(e_k)yyyc'(r_1)yc'(r_2)\cdots yc'(r_n)y,}$$

fica claro onde $c(\mathcal{T})$ termina devido ao prefixo yy de $c(w)$.

Exemplo

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

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

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 $c'(e_1) = x^2yxyx^3yxyx^3y$, enquanto que $\mathcal{T}$ é codificada por

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

Teorema [Turing], 1936]

Existe uma máquina de Turing $\mathcal{T}_U$ sobre o alfabeto $\{x, y\}$ tal que, para qualquer máquina de Turing $\mathcal{T} = (Q, A, T, \delta, i, f, \Delta)$ e qualquer palavra $w \in A^*$, "$\mathcal{T}_U$ simula o processamento de w por $\mathcal{T}$", isto é,

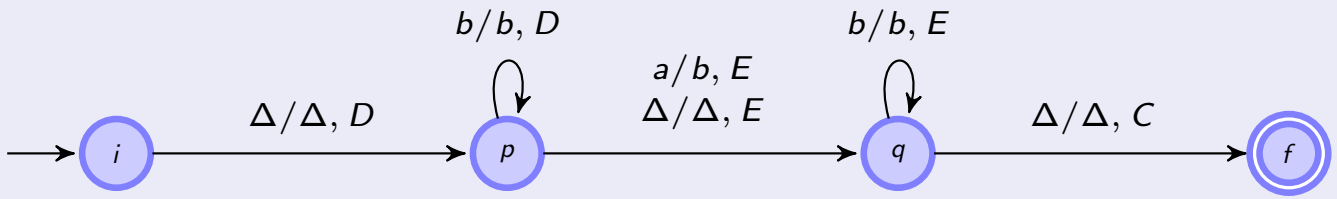
1. $\mathcal{T}$ aceita w se e só se $\mathcal{T}_U$ aceita $c(\mathcal{T})c(w)$. Neste caso, se a saída para w em $\mathcal{T}$ é u, então a saída para $c(\mathcal{T})c(w)$ em $\mathcal{T}_U$ é $c(u)$.
2. $\mathcal{T}$ entra em ciclo com w se e só se $\mathcal{T}_U$ entra em ciclo com $c(\mathcal{T})c(w)$.

Uma máquina de Turing $\mathcal{T}_U$ nas condições do teorema anterior é chamada uma *máquina de Turing universal*.

O comportamento de uma destas máquinas $\mathcal{T}_U$ será exemplificado com a máquina $\mathcal{T}$ do Exercício 1.27 (depois de resolvido o exercício).

Exercício 1.27

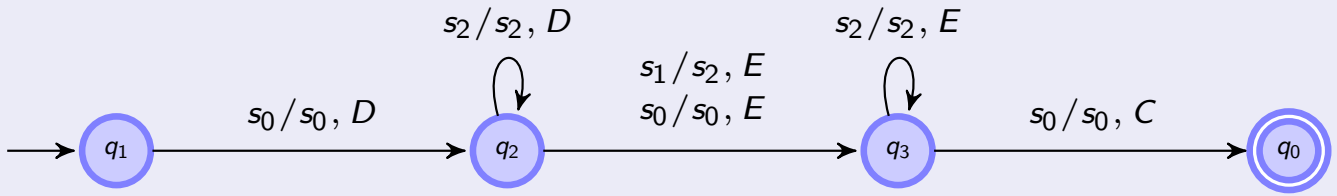
Seja $\mathcal{T}$ a máquina de Turing



que transforma uma dada palavra sobre o alfabeto $\{a, b\}$ numa outra em que a 1ª ocorrência da letra a (caso exista) é substituída por b. Codifique a máquina $\mathcal{T}$.

Resposta

Assumindo que $f = q_0$, $i = q_1$, $p = q_2$, $q = q_3$, $\Delta = s_0$, $a = s_1$, $b = s_2$, donde $\mathcal{T}$ é



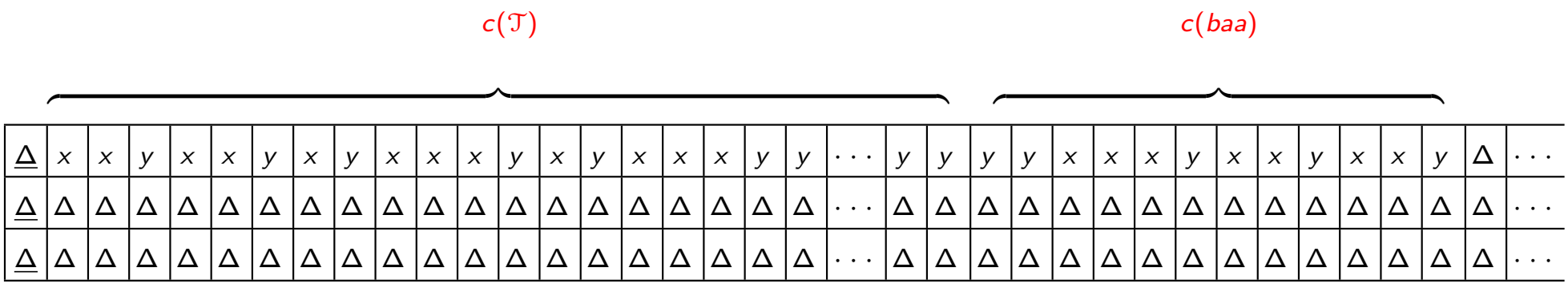
e que as transições são ordenadas da esquerda para a direita e de cima para baixo,

$$c(\mathcal{T}) = \overbrace{x^2}^{c'(q_1)}\, y\, \overbrace{x^2yxyx^3yxyx^3y}^{c'(e_1)}\, y\, \overbrace{x^3yx^3yx^3yx^3yx^3y}^{c'(e_2)}\, y\, \overbrace{x^3yx^2yx^4yx^3yx^2y}^{c'(e_3)}\, y$$
$$\underbrace{x^3yxyx^4yxyx^2y}_{c'(e_4)}\, y\, \underbrace{x^4yx^3yx^4yx^3yx^2y}_{c'(e_5)}\, y\, \underbrace{x^4yxyxyxyxy}_{c'(e_6)}\, y.$$

Simularemos o processamento da palavra $w = baa$ pela máquina $\mathcal{T}$.
Consideraremos uma máquina $\mathcal{T}_U$ com 3 fitas:

- a 1ª fita é a fita de entrada: é iniciada com a palavra $c(\mathcal{T})c(w)$. Esta fita é também a de saída: se $\mathcal{T}$ aceita w, então $\mathcal{T}_U$ aceita $c(\mathcal{T})c(w)$ e o conteúdo final da 1ª fita de $\mathcal{T}_U$ é o código do conteúdo final da fita de $\mathcal{T}$.
- a 2ª fita é a que simula o funcionamento de $\mathcal{T}$: contém o código do conteúdo da fita de $\mathcal{T}$ num dado instante.
- a 3ª fita contém o código do estado atual da máquina $\mathcal{T}$.

Note-se que $c(baa) = y^2x^3yx^2yx^2y$. As 3 fitas de $\mathcal{T}_U$ são portanto iniciadas como segue:



A máquina $\mathcal{T}_U$:

- começa por copiar $c(baa)$ para a $2^{\underline{a}}$ fita (exceto o prefixo yy), escrevendo ainda xy como prefixo para representar o símbolo Δ que está na $1^{\underline{a}}$ célula da fita de $\mathcal{T}$;
- depois copia o xx, que codifica o estado inicial de $\mathcal{T}$, para a $3^{\underline{a}}$ fita e apaga-o da $1^{\underline{a}}$ fita;
- coloca os cursores das 3 fitas na posição 2.

Δ	$\underline{x}$	x	y	x	y	x	x	x	y	x	y	x	x	x	y	y	$\cdots$	y	y	y	y	x	x	x	y	x	x	y	x	x	y	Δ	$\cdots$
Δ	$\underline{x}$	y	x	x	x	y	x	x	y	x	x	y	Δ	Δ	Δ	Δ	$\cdots$	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$
Δ	$\underline{x}$	x	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$

- A $3^{\underline{a}}$ fita indica que $\mathcal{T}$ está atualmente no estado q_1 e
- o cursor da $2^{\underline{a}}$ fita indica que o cursor da fita de $\mathcal{T}$ está a ler o símbolo Δ.

Assim, $\mathcal{T}_U$ vai procurar na $1^{\underline{a}}$ fita o quíntuplo (de potências de x, separadas pelo símbolo y) que codifica a transição cujas duas primeiras componentes correspondem a estes dados.

Neste exemplo, por acaso, o cursor da 1ª fita já está colocado na posição correspondente aos dados das outras fitas. Agora, o quíntuplo da 1ª fita diz que:

- na máquina $\mathcal{T}$ o novo estado passa a ser o q_2,
- o símbolo Δ deve ser mantido;
- o cursor deve mover-se para a direita.

Logo, a máquina $\mathcal{T}_U$ transforma as 3 fitas e o resultado é o seguinte:

Δ	$\underline{x}$	x	y	x	y	x	x	x	y	x	y	x	x	x	y	y	$\cdots$	y	y	y	y	x	x	x	y	x	x	y	x	x	y	Δ	$\cdots$
Δ	x	y	$\underline{x}$	x	x	y	x	x	y	x	x	y	Δ	Δ	Δ	Δ	$\cdots$	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$
Δ	$\underline{x}$	x	x	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$

Como resultado da próxima iteração, obter-se-ia

Δ	$\underline{x}$	x	y	x	y	x	x	x	y	x	y	x	x	x	y	y	$\cdots$	y	y	y	y	x	x	x	y	x	x	y	x	x	y	Δ	$\cdots$
Δ	x	y	x	x	x	y	$\underline{x}$	x	y	x	x	y	Δ	Δ	Δ	Δ	$\cdots$	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$
Δ	$\underline{x}$	x	x	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$

e a seguir

Δ	$\underline{x}$	x	y	x	y	x	x	x	y	x	y	x	x	x	y	y	$\cdots$	y	y	y	y	x	x	x	y	x	x	y	x	x	y	Δ	$\cdots$
Δ	x	y	$\underline{x}$	x	x	y	x	x	x	y	x	x	y	Δ	Δ	Δ	$\cdots$	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$
Δ	$\underline{x}$	x	x	x	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$

- O processo é iterado até que a máquina $\mathcal{T}_U$ detete, na 3ª fita, que a máquina $\mathcal{T}$ chegou ao estado final q_0.
- Então $\mathcal{T}_U$ copia o conteúdo da 2ª fita para a 1ª (depois de apagar esta) e vai para o seu estado final.

A situação final das 3 fitas seria a seguinte:

Δ	$\underline{x}$	y	x	x	x	y	x	x	x	y	x	x	y	Δ	$\cdots$
Δ	$\underline{x}$	y	x	x	x	y	x	x	x	y	x	x	y	Δ	$\cdots$
Δ	$\underline{x}$	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	$\cdots$

Portanto a situação final da fita da máquina $\mathcal{T}$ é $\underline{\Delta}bba$.

De seguida apresenta-se exemplos concretos de uma linguagem não recursivamente enumerável e de uma linguagem recursivamente enumerável que não é recursiva.

- Seja NãoAutoAceite, também denotado por NAA, o seguinte subconjunto de $\{x, y\}^*$:

$$\text{NãoAutoAceite} = NAA_1 \cup NAA_2$$

onde

$$\begin{aligned} NAA_1 &= \{w \in \{x,y\}^* \mid w = c(\mathcal{T}) \text{ e } \mathcal{T} \text{ não reconhece } w\} \\ NAA_2 &= \{w \in \{x,y\}^* \mid w \neq c(\mathcal{T}) \text{ para toda a máquina de Turing } \mathcal{T}\ \}. \end{aligned}$$

- Seja ainda AutoAceite o conjunto complementar

$$AA = \{x, y\}^* \setminus NAA$$

de NãoAutoAceite em $\{x, y\}^*$, ou seja,

$$\text{AutoAceite} = \{w \in \{x,y\}^* \mid w = c(\mathcal{T}) \text{ e } \mathcal{T} \text{ reconhece } w\}.$$

Teorema

NãoAutoAceite é uma linguagem não recursivamente enumerável.

Demonstração: Suponhamos que existe uma máquina de Turing $\mathcal{T}_0$ que aceita NAA, ou seja,

$$\mathcal{T}_0 \text{ aceita uma palavra } w \in \{x, y\}^* \text{ se e só se } w \in NAA. \tag{1}$$

Seja $w_0 = c(\mathcal{T}_0)$ e consideremos a questão: $\mathcal{T}_0$ aceita w_0?

- Se $\mathcal{T}_0$ aceita w_0, então $w_0 \in NAA$ por (1). Como $w_0 = c(\mathcal{T}_0)$ e $w_0 \in NAA$, então $w_0 \in NAA_1$. Logo $\mathcal{T}_0$ não aceita w_0, o que é uma contradição.
- Se $\mathcal{T}_0$ não aceita w_0, então $w_0 \in NAA_1$ e portanto $w_0 \in NAA$. Logo $\mathcal{T}_0$ aceita w_0 por (1), o que é uma contradição.

Em ambos os casos atinge-se uma contradição. Por isso, a máquina $\mathcal{T}_0$ não existe e NAA não é recursivamente enumerável. □

A seguir mostra-se que a linguagem AA é reconhecida por máquinas de Turing, mas cada máquina dessas entra em ciclo pelo menos para uma entrada $w \notin AA$.

Teorema

A linguagem AutoAceite é recursivamente enumerável mas não é recursiva.

Demonstração: Seja $w \in \{x, y\}^*$. Tem-se

$$\begin{aligned} w \in AA &\Leftrightarrow w = c(\mathcal{T}) \text{ e } \mathcal{T} \text{ aceita } w, \text{ para alguma máquina } \mathcal{T} \\ &\Leftrightarrow w = c(\mathcal{T}) \text{ e } \mathcal{T}_U \text{ aceita } c(\mathcal{T})c(w), \text{ onde } \mathcal{T}_U \text{ é uma MT universal} \\ &\Leftrightarrow w = c(\mathcal{T}) \text{ e } \mathcal{T}_U \text{ aceita } wc(w). \end{aligned}$$

Constrói-se uma máquina de Turing $\mathcal{T}_1$ (provar que existe) que:

- rejeite w se w não é código de alguma MT;
- deixe na fita $wc(w)$ se $w = c(\mathcal{T})$ para alguma máquina $\mathcal{T}$.

Então AA é reconhecida pela máquina $\mathcal{T}_1\mathcal{T}_U$. Portanto AA é recursivamente enumerável.

Que AA não é recursiva é consequência do teorema anterior. De facto, se AA fosse recursiva, então $\overline{AA} = NAA$ seria também recursiva. □