Capítulo 4

Expressões regulares

As referências à noção de expressão regular remontam a 1956, com Kleene, que a utilizou na descrição dos seus "conjuntos regulares". Para além da sua importância no domínio teórico das linguagens formais, as expressões regulares têm actualmente aplicação prática em diversas áreas, de entre as quais destacamos a dos sistemas operativos.

As expressões regulares são essenciais na construção de compiladores, no que concerne ao desenvolvimento de analisadores lexicais, bem como na pesquisa de informação em bases de dados, encontrando-se ainda integradas em diversos editores e processadores de texto como ferramenta de pesquisa de padrões.

Por exemplo, os sistemas Unix (Linux) disponibilizam a ferramenta GREP (Global search for Regular Expression and Print) a qual suporta a utilização de diversos operadores regulares na especificação de padrões de pesquisa, constituindo uma extensão do conceito original de expressão regular. Estas expressões regulares estão também na base de sistemas de geração automática de analisadores lexicais como o LEX ou o FLEX.

A seguinte tabela apresenta uma listagem de alguns operadores utilizados em GREP.

`*`	(zero ou mais ocorrências de ...)	`|`	(ou)
`+`	(uma ou mais ocorrências de ...)	`{ }`	(número de ocorrências de ...)
`?`	(zero ou uma ocorrência de ...)	`[ ]`	(uma ocorrência de ...)
`.`	(qualquer carácter)	`-`	(sub-domínio)

Os operadores anteriores permitem descrever sucintamente padrões complexos como os seguintes.

`a{3}`	aaa
`[0-9]+`	um inteiro sem sinal
`a{2,4}`	aa ou aaa ou aaaa
`[A-Za-z][a-zA-Z0-9]*`	um identificador

Por exemplo, a especificação de um número real pode ser a seguinte:

```
[+-]? [0-9]+ (\.[0-9]+)? ( (e|E) [+-]? [0-9]+ )?
```

Neste capítulo vamos formalizar a noção de expressão regular, estudar as propriedades das linguagens associadas às expressões regulares, bem como a relação destas com as linguagens reconhecidas com os autómatos finitos.

4.1 Linguagens das expressões regulares

Uma expressão regular é essencialmente uma especificação sucinta e formal de uma linguagem. Vamos considerar um modelo simples, baseado apenas nos operadores de *adição* (união de linguagens), *produto* (concatenação de linguagens) e *estrela de Kleene* (fecho de Kleene).

A cada expressão regular E associamos uma linguagem denotada por $\mathcal{L}(E)$. As linguagens que podem ser especificadas por intermédio de expressões regulares são designadas por *linguagens regulares*.

Definição 4.1.1 Expressão Regular.

Seja $a \in \Sigma$ e sejam E e F expressões regulares. Uma *expressão regular* é definida indutivamente pelos seguintes casos:

Base (operandos).

Expressão (sintaxe)	Linguagem (semântica)
$\emptyset$	$\mathcal{L}(\emptyset) = \emptyset$
ε	$\mathcal{L}(\varepsilon) = \{\varepsilon\}$
a	$\mathcal{L}(a) = \{a\}$

Indução (operadores).

Expressão (sintaxe)	Linguagem (semântica)
(E)	$\mathcal{L}((E)) = \mathcal{L}(E)$
$(E + F)$	$\mathcal{L}(E + F) = \mathcal{L}(E) \cup \mathcal{L}(F)$
(EF)	$\mathcal{L}(EF) = \mathcal{L}(E)\mathcal{L}(F)$
(E^*)	$\mathcal{L}(E^*) = \mathcal{L}(E)^\star$

Definição 4.1.2 Linguagem Regular.

Uma **linguagem regular** (LR) é uma linguagem associada a alguma expressão regular.

Duas expressões regulares são iguais sempre que as linguagens regulares associadas também o sejam.

Os parêntesis nas expressões regulares têm como objectivo indicar a ordem pela qual as operações são efectuadas. Para evitar o seu excesso, convencionamos as seguintes *precedências dos operadores*:

1º	$*$ (estrela de Kleene)
2º	(produto)
3º	$+$ (adição)

Por exemplo, aplicando as precedências anteriores à expressão regular $a + b^*c$, obtemos

$$a + b^*c = \big((a) + \big(((b)^*)\,(c)\big)\big).$$

Sejam R uma expressão regular e $n \in \mathbb{N}$. Denotamos por R^n a concatenação iterada, n vezes, de R:

$$R^n = \underbrace{RR\cdots R}_{n \text{ factores}}$$

e convencionamos que $R^0 = \varepsilon$.

Utilizamos o símbolo de somatório para denotar a soma iterada de expressões regulares:

$$\sum_{i=1}^{n} R_i = \underbrace{R_1 + R_2 + \cdots + R_n}_{n \text{ parcelas}}.$$

As expressões regulares não são mais do que especificações formais de linguagens. Sem grande surpresa, verificamos que muitas propriedades algébricas das expressões regulares decorrem das propriedades das linguagens associadas. A título de exemplo, apresentamos a demonstração da propriedade associativa do produto de expressões regulares.

Teorema 4.1.3

Se E, F e G são expressões regulares quaisquer, então

$$E(FG) = (EF)G.$$

Demonstração.

Pela definição de produto de expressões regulares,

$$\mathcal{L}(E(FG)) = \mathcal{L}(E)\mathcal{L}(FG) = \mathcal{L}(E)(\mathcal{L}(F)\mathcal{L}(G)).$$

Uma vez que a concatenação de linguagens é uma operação associativa, podemos escrever que

$$\mathcal{L}(E)(\mathcal{L}(F)\mathcal{L}(G)) = (\mathcal{L}(E)\mathcal{L}(F))\mathcal{L}(G).$$

Usando uma vez mais a definição de produto de expressões regulares, obtemos

$$(\mathcal{L}(E)\mathcal{L}(F))\mathcal{L}(G) = \mathcal{L}(EF)\mathcal{L}(G) = \mathcal{L}((EF)G).$$

Exemplo 4.1.4

São exemplos de linguagens regulares especificadas por intermédio de expressões regulares:

- Expressão Regular: $(0+1)^*$

  Linguagem: $\mathcal{L}((0+1)^*) = (\{0\} \cup \{1\})^\star = \{0,1\}^\star$

  (todas as palavras com letras no alfabeto binário)

- Expressão Regular: $0 + 10^*$

  Linguagem: $\{0\} \cup \{1\}\{0\}^\star = \{0, 1, 10, 100, 1000, \ldots\}$

  (a palavra 0, a palavra 1 e todas as palavras começadas por 1 e seguidas de um ou mais zeros)

- Expressão Regular: $(0+1)^*(0+11)$

  Linguagem: $\{0,1\}^\star\{0,11\}$

  (todas as palavras terminadas em 0 ou em 11)

- Expressão Regular: $(0+1)^*011(0+1)^*$

  Linguagem: $\{0,1\}^\star\{011\}\{0,1\}^\star$

  (todas as palavras que contêm a sub-palavra 011)

- Expressão Regular: $(10)^* + (01)^* + 0(10)^* + 1(01)^*$

  Linguagem: todas as palavras com 0's e 1's alternados

- Expressão Regular: $(\varepsilon + 1)(01)^*(\varepsilon + 0)$

  Linguagem: todas as palavras com 0's e 1's alternados

- Expressão Regular: $(\varepsilon + 0)(1 + 10)^*$

  Linguagem: todas as palavras sem 0's consecutivos

- Expressão Regular: $(1 + 01)^*(\varepsilon + 0)$

  Linguagem: todas as palavras sem 0's consecutivos

4.2 Expressões regulares e autómatos

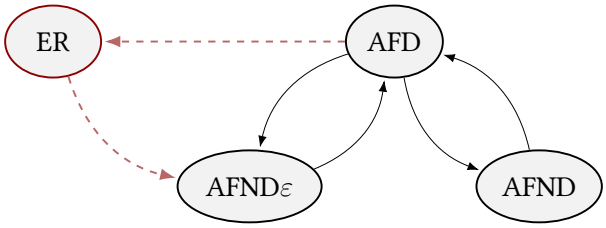
No Capítulo 2, p. 29, constatámos que os três tipos de autómatos finitos aí definidos permitem especificar linguagens (as linguagens reconhecidas pelos autóma-

tos). Demonstrámos que as classes das linguagens reconhecidas por autómatos finitos não deterministas (AFND e AFNDε) e das linguagens reconhecidas pelos autómatos finitos deterministas (AFD) são idênticas.

Na secção anterior vimos que as expressões regulares providenciam uma descrição formal sucinta de linguagens, as quais constituem a *classe das linguagens regulares*.

Vamos verificar que a classe de linguagens reconhecidas por autómatos finitos é idêntica à classe das linguagens regulares, pelo que podemos especificar as linguagens regulares com expressões regulares ou com autómatos finitos.

O esquema seguinte ilustra todas estas relações, assinalando a tracejado as inclusões que vamos demonstrar.



4.2.1 Conversão ER para AFNDε

O *Teorema de Thompson* (Ken Thompson, Estados Unidos da América, 1943 -) afirma que é sempre possível definir um autómato finito equivalente a uma expressão regular, no sentido em que a linguagem reconhecida pelo autómato é a mesma que a linguagem especificada pela expressão.

Para além da existência de um AFNDε equivalente a uma dada expressão regular, a demonstração do teorema fornece um método indutivo para o construir.

Teorema 4.2.1 Teorema de Thompson.

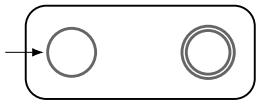
Se R é uma expressão regular então existe um AFNDε *E tal que $\mathcal{L}(R) = \mathcal{L}(E)$.*

Demonstração.

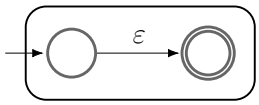
Vamos construir um AFNDε para cada forma possível de expressão regular seguindo a definição indutiva. Cada um dos autómatos construídos possui um só estado de aceitação e é diferente do estado inicial.

Casos Base.

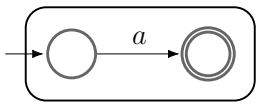
- Se $R = \emptyset$ então $\mathcal{L}(R) = \emptyset$ e o autómato E ilustrado na seguinte figura satisfaz $\mathcal{L}(R) = \mathcal{L}(E)$:



- Se $R = \varepsilon$ então $\mathcal{L}(R) = \{\varepsilon\}$ e o autómato E ilustrado na seguinte figura satisfaz $\mathcal{L}(R) = \mathcal{L}(E)$:



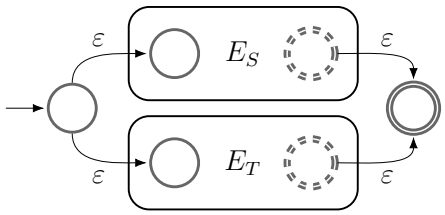
- Se $R = a$ com $a \in \Sigma$ então $\mathcal{L}(R) = \{a\}$ e o autómato E ilustrado na seguinte figura satisfaz $\mathcal{L}(R) = \mathcal{L}(E)$:



Indução.

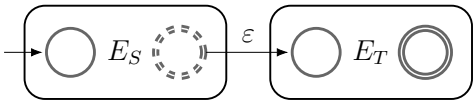
- Se S e T são expressões regulares então a linguagem associada à expressão regular $R = S + T$ é $\mathcal{L}(R) = \mathcal{L}(S) \cup \mathcal{L}(T)$. Por hipótese de indução, existem autómatos finitos não deterministas com transições ε, E_S e E_T, tais que $\mathcal{L}(S) = \mathcal{L}(E_S)$ e $\mathcal{L}(T) = \mathcal{L}(E_T)$. Uma vez que a classe dos autómatos finitos não deterministas com transições ε é fechada para a união de linguagens, existe um AFNDε E_R tal que $\mathcal{L}(E_R) = \mathcal{L}(E_S) \cup \mathcal{L}(E_T) = \mathcal{L}(R)$.

  A construção do autómato E_R a partir dos autómatos E_S e E_T é:



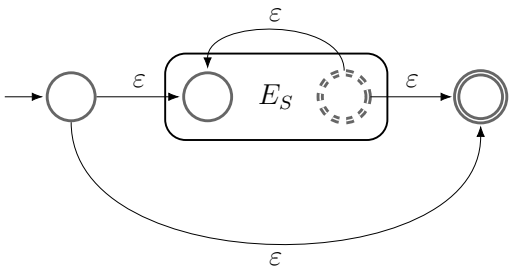
- Se S e T são expressões regulares então a linguagem associada à expressão regular $R = ST$ é $\mathcal{L}(R) = \mathcal{L}(S)\mathcal{L}(T)$. Por hipótese de indução, existem autómatos finitos não deterministas com transições ε, E_S e E_T, tais que $\mathcal{L}(S) = \mathcal{L}(E_S)$ e $\mathcal{L}(T) = \mathcal{L}(E_T)$. Uma vez que a classe dos autómatos finitos não deterministas com transições ε é fechada para concatenação de linguagens, existe um AFNDε E_R tal que $\mathcal{L}(E_R) = \mathcal{L}(E_S)\mathcal{L}(E_T) = \mathcal{L}(R)$.

  A construção do autómato E_R é:

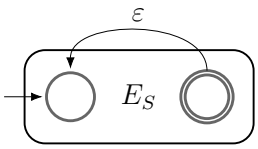


- Se S é uma expressão regular então a linguagem associada à expressão regular $R = S^*$ é $\mathcal{L}(R) = \mathcal{L}(S)^\star$. Por hipótese de indução, existe um autómato finito não determinista com transições ε, E_S, tal que $\mathcal{L}(S) = \mathcal{L}(E_S)$. Uma vez que a classe dos autómatos finitos não deterministas com transições ε é fechada para o fecho de Kleene, existe um AFNDε E_R tal que $\mathcal{L}(E_R) = \mathcal{L}(E_S)^\star = \mathcal{L}(R)$.

  A construção do autómato E_R é:



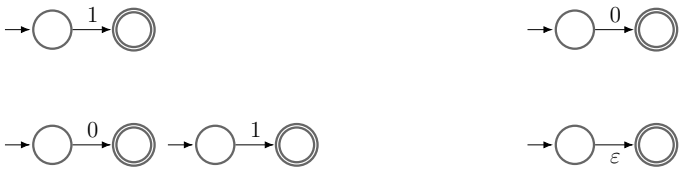
Atenção! A linguagem associada ao esquema seguinte não é $\mathcal{L}(S)^\star$, mas sim $\mathcal{L}(S)^+$.



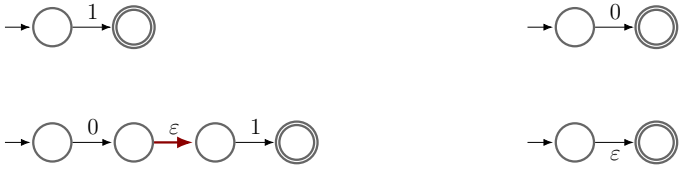
Exemplo 4.2.2

Aplicando o método descrito na demonstração do teorema Teorema de Thompson, p. 89, construímos um AFNDε reconhecedor de $\mathcal{L}(R)$, onde $R = (1{+}01)^*(0{+}\varepsilon)$.

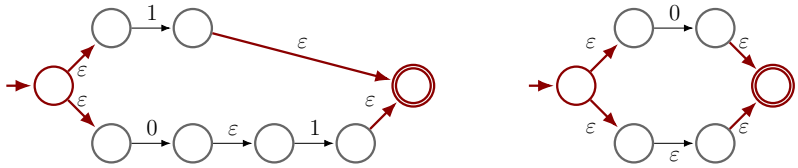
1. Construção dos autómatos associados às 5 expressões regulares 1, 0, 1, 0 e ε:



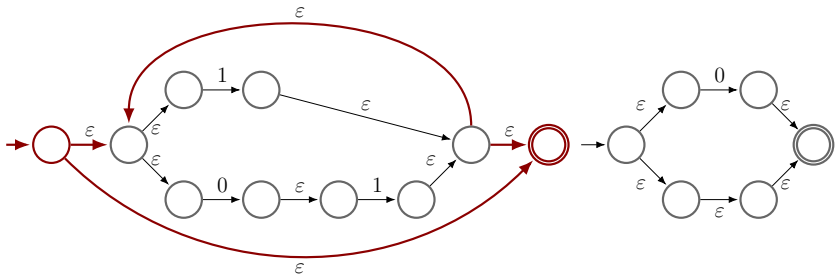
2. Operação produto das expressões regulares 0 com 1:



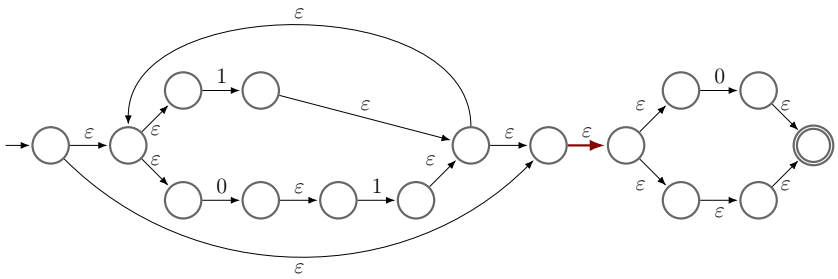
3. Operação adição das expressões regulares 1 com 01 e das expressões regulares 0 com ε:



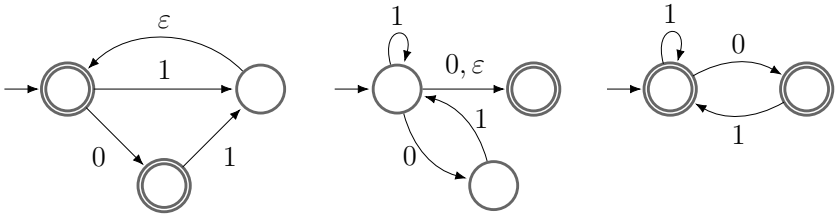
4. Operação estrela de Kleene da expressão regular $1 + 01$:



5. Operação produto das expressões regulares $(1+01)^*$ com $0+\varepsilon$:



É claro que existem outros autómatos reconhecedores de $\mathcal{L}(R)$, por exemplo:



4.2.2 Conversão de AFD para ER

Vamos analisar agora o problema da construção de uma expressão regular que representa a linguagem reconhecida por um autómato finito determinista. O *Teorema de Kleene* garante a existência dessas expressões regulares.

Teorema 4.2.3 Teorema de Kleene.

Se $\mathcal{L}(A)$ é a linguagem reconhecida por um autómato finito determinista A então existe uma expressão regular R tal que $\mathcal{L}(A) = \mathcal{L}(R)$.

Demonstração.
Seja $Q = \{1, 2, \ldots, n\}$ o conjunto de estados de um AFD A.

Para $i, j \in Q$, o conjunto das palavras que resultam de percorrer todos os passeios no diagrama de transições do autómato A com início no estado i, que não passam por estados intermédios maiores do que k e que terminam no estado j é uma linguagem que denotamos por $\mathcal{L}_{ij}^{(k)}$. Por exemplo:

- $\mathcal{L}_{ij}^{(0)}$ é a linguagem das palavras que resultam de transições simples de i para j.
- $\mathcal{L}_{ij}^{(n)}$ é a linguagem de todas as palavras que resultam de transições estendidas de i para j.

Vamos demonstrar por indução sobre k que $\mathcal{L}_{ij}^{(k)}$ é representável por uma expressão regular que denotamos por $R_{ij}^{(k)}$.

Caso base. Para $k = 0$, há dois casos a considerar:

Se $i \neq j$ então

- ou não há qualquer transição directa de i para j e nesse caso $R_{ij}^{(0)} = \emptyset$;
- ou existem transições directas de i para j pelos símbolos $a_1, a_2, \ldots, a_l$ e nesse caso $R_{ij}^{(0)} = a_1 + a_2 + \cdots + a_l$.

Se $i = j$ então

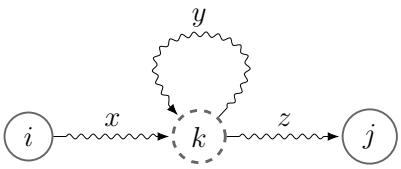
- ou não há qualquer transição directa de i para i e nesse caso $R_{ii}^{(0)} = \varepsilon$;
- ou existem transições directas de i para i pelos símbolos $a_1, a_2, \ldots, a_l$ e nesse caso $R_{ii}^{(0)} = \varepsilon + a_1 + a_2 + \cdots + a_l$.

Passo indutivo. Admitindo que a linguagem $\mathcal{L}_{ij}^{(k-1)}$ é representável por uma expressão regular $R_{ij}^{(k-1)}$, quaisquer que sejam $i, j \in Q$, vamos provar que $\mathcal{L}_{ij}^{(k)}$ é também representável por uma expressão regular $R_{ij}^{(k)}$.

Consideremos o passeio no diagrama de transições associado a uma qualquer palavra w da linguagem $\mathcal{L}_{ij}^{(k)}$.

Se o estado k não pertence a este passeio então $w \in \mathcal{L}_{ij}^{(k-1)}$ e portanto w é uma das palavras da linguagem representada pela expressão regular $R_{ij}^{(k-1)}$.

Se o estado k pertence a esse passeio então ele começa com um passeio do estado i até ao estado k, seguido de um passeio alternativo de uma ou mais voltas em torno do estado k e terminando com um passeio do estado k até ao estado j, conforme ilustramos na seguinte figura:



Observamos que:

- o passeio de i para k não passa por estados maiores do que $k-1$ e portanto a palavra associada, x, pertence à linguagem $\mathcal{L}_{ik}^{(k-1)}$;
- o passeio de k para j não passa por estados maiores do que $k-1$ e portanto a palavra associada, z, pertence à linguagem $\mathcal{L}_{kj}^{(k-1)}$;
- o ciclo alternativo de k para k não passa por estados maiores do que $k-1$ e portanto a palavra associada, y, pertence à linguagem $\mathcal{L}_{kk}^{(k-1)}$.

Assim, para algum $n \in \mathbb{N}_0$, $w = xy^n z$ é uma palavra da linguagem representada pela expressão regular $R_{ik}^{(k-1)} \left(R_{kk}^{(k-1)}\right)^* R_{kj}^{(k-1)}$.

Em qualquer dos casos, concluímos que w é uma palavra da linguagem $L_{ij}^{(k)}$ representada pela expressão regular

$$R_{ij}^{(k)} = R_{ij}^{(k-1)} + R_{ik}^{(k-1)} \left(R_{kk}^{(k-1)}\right)^* R_{kj}^{(k-1)}.$$

Finalmente, a expressão regular associada à linguagem do autómato A é a união $(+)$ de todas as expressões regulares da forma $R_{sf}^{(n)}$, do estado inicial s para um estado de aceitação f.

A construção indutiva das expressões regulares $R_{ij}^{(k)}$ definidas na demonstração do teorema anterior está na base de um método expedito para obter uma expressão regular representante da linguagem reconhecida por um autómato finito.

A ideia principal do algoritmo seguinte consiste em remover sucessivamente os estados internos do autómato, associando expressões regulares às transições directas entre estados. Quando o algoritmo termina, o autómato será constituído por apenas dois estados (o estado inicial e um estado de aceitação) e a expressão regular que rotula a única transição entre o estado inicial e o estado de aceitação especifica a linguagem reconhecida pelo autómato original.

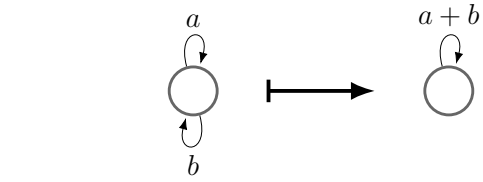
Notamos que este algoritmo se pode aplicar a qualquer dos tipos de autómatos finitos estudados (AFD, AFND ou AFNDε).

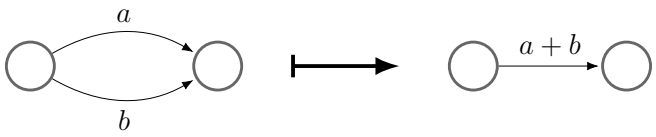
Algoritmo 4.2.4 Método de Eliminação de Estados.

Entrada: um autómato finito A.

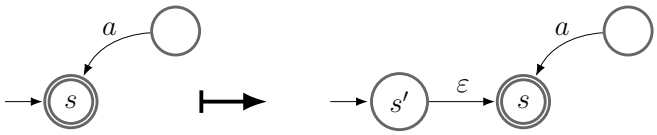
Saída: uma expressão regular equivalente.

1. Substituir todas as transições múltiplas por uma única transição:

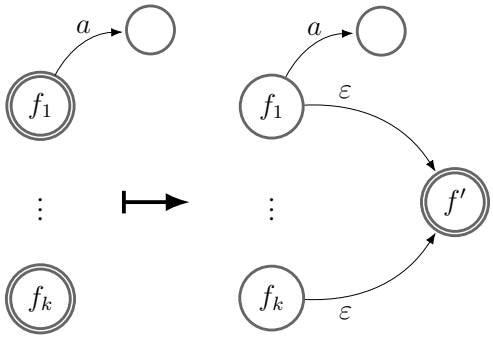




2. Se o estado inicial do autómato, s, é também um estado de aceitação ou se existem transições para s então criar um novo estado inicial s' e uma transição ε de s' para s:

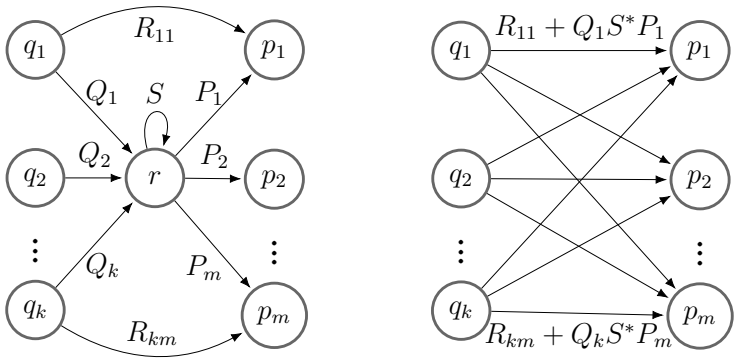


3. Se o autómato possui mais do que um estado de aceitação ou se existem transições a partir de um estado de aceitação então:
   - criar um novo estado de aceitação, f';
   - definir uma transição ε de cada um dos estados de aceitação originais para o novo estado de aceitação, f';
   - os estados de aceitação originais passam a estados de rejeição.



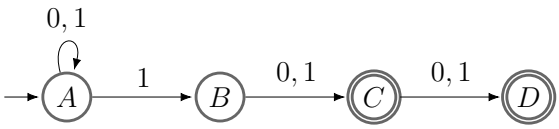
4. Para cada estado interno r:
   - Sejam $\{q_1, q_2, \ldots, q_k\}$ os estados incidentes em r.
   - Sejam $\{p_1, p_2, \ldots, p_m\}$ os estados emergentes de r.
   - Para $i = 1, 2, \ldots, k$, seja Q_i a expressão regular no arco de q_i para r.
   - Para $j = 1, 2, \ldots, m$, seja P_j a expressão regular no arco de r para p_j.
   - Para $i = 1, 2, \ldots, k$ e $j = 1, 2, \ldots, m$, seja R_{ij} a expressão regular no arco de q_i para p_j.

   Remover r, etiquetando cada arco de q_i para p_j pela expressão regular $R_{ij} + Q_i S^* P_j$ conforme ilustrado nas seguintes figuras:

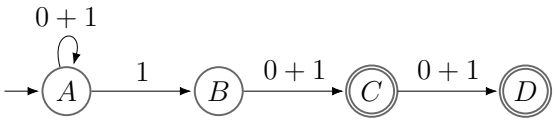


Exemplo 4.2.5

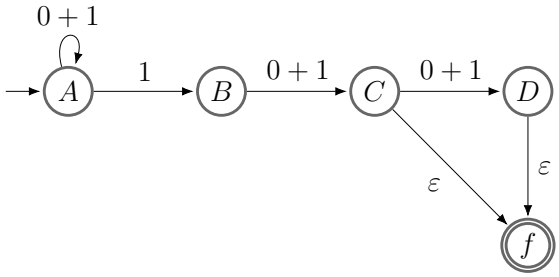
Consideremos o AFND reconhecedor das palavras binárias com o dígito 1 na penúltima ou ante-penúltima posição:



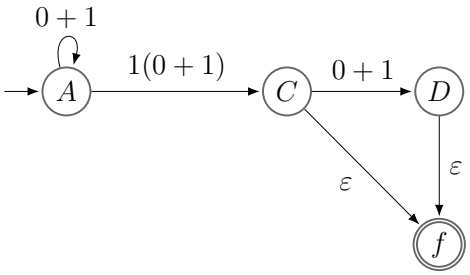
- Conversão das transições múltiplas:



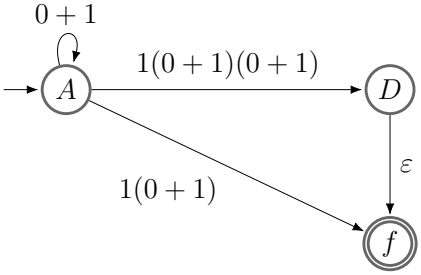
- Redução a um único estado de aceitação:



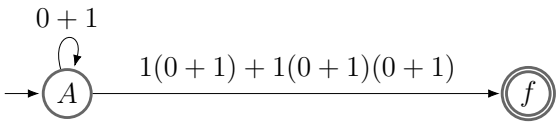
- Eliminação do estado interno B:



- Eliminação do estado interno C:



- Eliminação do estado interno D:



Assim, uma expressão regular que representa a linguagem das palavras binárias com o dígito 1 na penúltima ou ante-penúltima posição é

$$R = (0+1)^*\big(1(0+1) + 1(0+1)(0+1)\big).$$

4.3 Operações regulares

Nesta secção debruçamo-nos sobre a álgebra das expressões regulares bem como sobre as operações para as quais a classe das linguagens regulares é fechada. Começamos com uma listagem de algumas regras algébricas fundamentais.

4.3.1 Propriedades Algébricas

Associatividade e Comutatividade. A soma e o produto de expressões regulares são operações associativas. A adição é uma operação comutativa, mas o produto não é comutativo em geral:

$$\begin{aligned}
&R(ST) = (RS)T = RST \\
&R + (S + T) = (R + S) + T = R + S + T \\
&R + S = S + R \\
&\text{geralmente } RS \neq SR.
\end{aligned}$$

Distributividade. São válidas as propriedades distributivas do produto em relação à adição bem como da adição em relação ao produto:

$$\begin{aligned}
R(S + T) &= RS + RT \\
(R + S)T &= RT + ST.
\end{aligned}$$

Elementos Neutro e Absorvente. A expressão regular ε é o elemento neutro do produto de expressões regulares, enquanto a expressão regular $\emptyset$ é o elemento absorvente do produto e simultaneamente o elemento neutro da adição:

$$\begin{aligned}
&\varepsilon R = R\varepsilon = R \\
&\emptyset + R = R + \emptyset = R \\
&\emptyset R = R\emptyset = \emptyset.
\end{aligned}$$

Leis da Estrela de Kleene.

$$(R^*)^* = R^*$$
$$\varepsilon^* = \varepsilon$$
$$\emptyset^* = \varepsilon$$
$$R^+ = RR^* = R^*R$$
$$R^* = R^+ + \varepsilon.$$

Idempotência.

$$R + R = R.$$

Em termos algébricos, as propriedades anteriores conjuntamente com um conjunto de axiomas de ordem, derivados da ordem parcial entre expressões regulares induzida pela relação de inclusão ($\subseteq$) entre expressões regulares, permite conferir à classe das expressões regulares a estrutura de uma Álgebra de Kleene, [16], [7], [15].

Com o objectivo de ilustrar as técnicas de demonstração de propriedades que envolvam expressões regulares, vamos provar a propriedade da distributividade à esquerda do produto relativamente à adição de expressões regulares.

Teorema 4.3.1

Se R, S e T são expressões regulares então $R(S + T) = RS + RT$.

Demonstração.

Vamos provar que as expressões regulares do lado esquerdo e do lado direito da igualdade enunciada estão associadas à mesma linguagem, ou seja, que

$$\mathcal{L}(R(S + T)) = \mathcal{L}(RS + RT),$$

ou ainda, usando a definição de ER, que

$$\mathcal{L}(R)\big(\mathcal{L}(S) \cup \mathcal{L}(T)\big) = \mathcal{L}(R)\mathcal{L}(S) \cup \mathcal{L}(R)\mathcal{L}(T).$$

1. $\mathcal{L}(R)\big(\mathcal{L}(S) \cup \mathcal{L}(T)\big) \subseteq \mathcal{L}(R)\mathcal{L}(S) \cup \mathcal{L}(R)\mathcal{L}(T)$:

   Seja $w \in \mathcal{L}(R)\big(\mathcal{L}(S) \cup \mathcal{L}(T)\big)$. Então $w = xy$, com $x \in \mathcal{L}(R)$ e $y \in \mathcal{L}(S) \cup \mathcal{L}(T)$.

   Assim, $x \in \mathcal{L}(R)$ e $y \in \mathcal{L}(S)$, donde $w \in \mathcal{L}(R)\mathcal{L}(S)$, ou então $x \in \mathcal{L}(R)$ e $y \in \mathcal{L}(T)$, donde $w \in \mathcal{L}(R)\mathcal{L}(T)$.

   Em qualquer dos casos, concluímos que $w \in \mathcal{L}(R)\mathcal{L}(S) \cup \mathcal{L}(R)\mathcal{L}(T)$.

2. $\mathcal{L}(R)\mathcal{L}(S) \cup \mathcal{L}(R)\mathcal{L}(T) \subseteq \mathcal{L}(R)\big(\mathcal{L}(S) \cup \mathcal{L}(T)\big)$:

   Seja $w \in \mathcal{L}(R)\mathcal{L}(S) \cup \mathcal{L}(R)\mathcal{L}(T)$. Então $w \in \mathcal{L}(R)\mathcal{L}(S)$ ou $w \in \mathcal{L}(R)\mathcal{L}(T)$.

No primeiro caso, $w = xy$, com $x \in \mathcal{L}(R)$ e $y \in \mathcal{L}(S)$. Uma vez que $\mathcal{L}(S) \subseteq \mathcal{L}(S) \cup \mathcal{L}(T)$, $y \in \mathcal{L}(S) \cup \mathcal{L}(T)$. Logo, $w \in \mathcal{L}(R)\big(\mathcal{L}(S) \cup \mathcal{L}(T)\big)$.

No segundo caso, $w = xy$, com $x \in \mathcal{L}(R)$ e $y \in \mathcal{L}(T)$. Uma vez que $\mathcal{L}(T) \subseteq \mathcal{L}(S) \cup \mathcal{L}(T)$, $y \in \mathcal{L}(S) \cup \mathcal{L}(T)$. Logo, $w \in \mathcal{L}(R)\big(\mathcal{L}(S) \cup \mathcal{L}(T)\big)$.

Se uma linguagem L_1 é sub-linguagem de uma linguagem L_2, $L_1 \subseteq L_2$, então $L_1^n \subseteq L_2^n$, para qualquer $n \in \mathbb{N}_0$, e ainda $L_1^\star \subseteq L_2^\star$. Ao aplicar estas propriedades às linguagens associadas às expressões regulares, estabelecemos os resultados seguintes.

Lema 4.3.2

Sejam R e S expressões regulares tais que $\mathcal{L}(R) \subseteq \mathcal{L}(S)$. Então, para $n \in \mathbb{N}_0$, $\mathcal{L}(R^n) \subseteq \mathcal{L}(S^n)$.

Lema 4.3.3

Sejam R e S expressões regulares tais que $\mathcal{L}(R) \subseteq \mathcal{L}(S)$. Então $\mathcal{L}(R^*) \subseteq \mathcal{L}(S^*)$.

4.3.2 Operações com Linguagens Regulares

A classe das linguagens regulares é fechada para as operações de *união*, *intersecção*, *concatenação*, *reversão*, *diferença*, *complementar* e *fecho de Kleene*.

Teorema 4.3.4

Se L_1 e L_2 são linguagens regulares então a união $L_1 \cup L_2$ *é uma linguagem regular.*

Demonstração.
Sejam R_1 e R_2 expressões regulares associadas às linguagens regulares L_1 e L_2, respectivamente. Então $L_1 \cup L_2$ é uma linguagem regular, pois $L_1 \cup L_2 = \mathcal{L}(R_1 + R_2)$, por definição de expressão regular.

Teorema 4.3.5

Se L é uma linguagem regular então o complementar $\overline{L} = \Sigma^* \setminus L$ *é uma linguagem regular.*

Demonstração.
Se $A = (Q, \Sigma, \delta, s, T)$ é um AFD completo que reconhece L então

$$B = (Q, \Sigma, \delta, s, Q \setminus T)$$

é um AFD que reconhece $\overline{L}$. Logo, $\overline{L}$ é regular.

Teorema 4.3.6

Se L_1 e L_2 são linguagens regulares então a intersecção $L_1 \cap L_2$ *é uma linguagem regular.*

Demonstração.

$L_1 \cap L_2$ é uma linguagem regular pois $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$ e a classe das linguagens regulares é fechada para a união de linguagens e para o complementar de uma linguagem.

Teorema 4.3.7

Se L_1 e L_2 são linguagens regulares então a diferença $L_1 \setminus L_2$ *é uma linguagem regular.*

Demonstração.

$L_1 \setminus L_2$ é uma linguagem regular pois $L_1 \setminus L_2 = L_1 \cap \overline{L_2}$ e a classe das linguagens regulares é fechada para a intersecção de linguagens e para o complementar de uma linguagem.

Teorema 4.3.8

Se L_1 e L_2 são linguagens regulares então a concatenação $L_1 L_2$ *é uma linguagem regular.*

Demonstração.

Sejam R_1 e R_2 expressões regulares associadas às linguagens regulares L_1 e L_2, respectivamente. Então $L_1 L_2$ é uma linguagem regular uma vez que $L_1 L_2 = \mathcal{L}(R_1 R_2)$, por definição de expressão regular.

Teorema 4.3.9

Se L é uma linguagem regular então o fecho de Kleene $L^\star$ *é uma linguagem regular.*

Demonstração.

Seja S uma expressão regular associada à linguagem regular L. Então $L^\star$ é uma linguagem regular uma vez que $L^\star = \mathcal{L}(S^*)$, por definição de expressão regular.

Relembramos que a linguagem reversa de uma linguagem L (Definição 1.2.12, p. 12) é o conjunto das palavras reversas das palavras em L. Dada uma expressão regular S denotamos por $\text{REV}(S)$ uma expressão regular que especifica a linguagem reversa da linguagem associada à expressão S, isto é,

$$\mathcal{L}(\text{REV}(S)) = \mathcal{L}(S)^{-1}.$$

Demonstramos agora que uma expressão regular tem sempre uma expressão reversa.

Teorema 4.3.10

A classe das linguagens regulares é fechada para a operação de reversão.

Demonstração.
Basta verificar que é possível reverter qualquer expressão regular, usando as definições de inversa e de expressão regular:

1. $\text{REV}(\emptyset) = \emptyset$;
2. para $a \in \Sigma \cup \{\varepsilon\}$, $\text{REV}(a) = a$;
3. se S e T são expressões regulares então $\text{REV}(S+T) = \text{REV}(S) + \text{REV}(T)$;
4. se S e T são expressões regulares então $\text{REV}(ST) = \text{REV}(T)\,\text{REV}(S)$;
5. se S é uma expressão regular então $\text{REV}\,(S^*) = \text{REV}(S)^*$.

Observação 4.3.11 Uma demonstração alternativa do teorema anterior consiste em começar por construir um autómato finito A (AFD, AFND, ou AFNDε) que reconheça a linguagem associada a S (por exemplo usando o algoritmo de Thompson).

Para obter um autómato finito A^{-1} que reconheça $\mathcal{L}(S)^{-1}$ basta:

1. inverter o sentido de todas as transições em A;
2. criar um novo estado inicial para A^{-1}, com transições ε para os estados de aceitação do autómato A;
3. fazer do estado inicial original de A o (único) estado de aceitação de A^{-1}.

Finalmente, construímos uma expressão regular para $\text{REV}(S)$ aplicando o algoritmo de Kleene ao autómato A^{-1}.

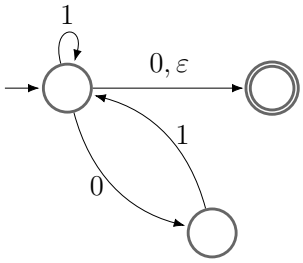
Exemplo 4.3.12

Consideremos a expressão regular $S = (1+01)^*(0+\varepsilon)$.

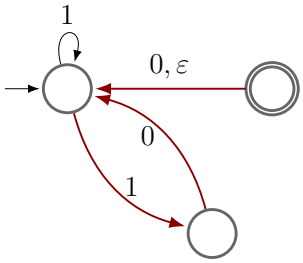
Seguindo a demonstração do Teorema 4.3.10, p. 103, uma expressão regular para a linguagem reversa da linguagem definida por S é dada por

$$\begin{aligned}
\text{REV}(S) &= \text{REV}\big((1+01)^*(0+\varepsilon)\big) \\
&= \text{REV}\big(0+\varepsilon\big)\,\text{REV}\big((1+01)^*\big) \\
&= \text{REV}\big(0+\varepsilon\big)\big(\text{REV}(1+01)\big)^* \\
&= \big(\text{REV}(0)+\text{REV}(\varepsilon)\big)\big(\text{REV}(1)+\text{REV}(01)\big)^* \\
&= (0+\varepsilon)(1+10)^*.
\end{aligned}$$

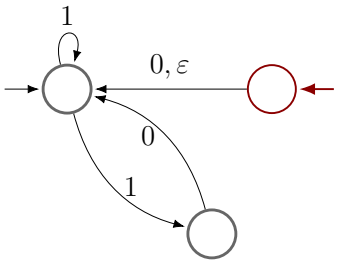
De modo alternativo, começamos por construir um AFND reconhecedor da linguagem associada a S:



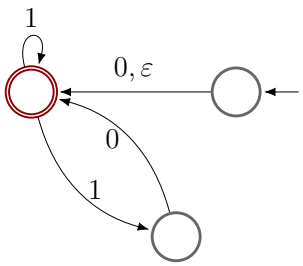
1 - Invertemos os sentidos de todas as transições



2 - Transformamos o único estado de aceitação do autómato original no estado inicial



3 - Transformamos o estado inicial do autómato original no único estado de aceitação



Por fim, basta encontrar uma expressão regular para a linguagem deste autómato (deixamos como exercício).

4.3.3 Homomorfismos regulares

Se R é uma expressão regular sobre um alfabeto Σ e $h\colon \Sigma^\star \to \Gamma^\star$ é um homomorfismo então $h(R)$ é a expressão regular sobre Γ obtida por substituição em R de cada símbolo $a \in \Sigma$ pela palavra (encarada como uma expressão regular) $h(a) \in \Gamma^\star$.

Lema 4.3.13

Se R é uma expressão regular sobre um alfabeto Σ e $h\colon \Sigma^\star \to \Gamma^\star$ é um homomorfismo então $\mathcal{L}(h(R)) = h\big(\mathcal{L}(R)\big)$.

Demonstração.

Decorre da definição que $h(\varepsilon) = \varepsilon$, $h(S+T) = h(S)+h(T)$, $h(ST) = h(S)h(T)$ e $h(S^*) = h(S)^*$, quaisquer que sejam as expressões regulares S e T.

A propriedade enunciada demonstra-se com indução estrutural sobre a definição de expressão regular, aplicando ainda as propriedades da imagem directa de um conjunto por intermédio de um homomorfismo e a definição de linguagem associada a uma expressão regular. Deixamos os detalhes ao cuidado do leitor, como exercício.

A linguagem imagem de uma linguagem regular por intermédio de um homomorfismo é ainda uma linguagem regular. Este resultado é um caso particular do teorema da substituição que apresentaremos mais à frente. Assim ***a classe das linguagens regulares é fechada para homomorfismos***.

Teorema 4.3.14

Seja L uma linguagem regular sobre um alfabeto Σ e seja $h\colon \Sigma^\star \to \Gamma^\star$ um homomorfismo. Então $h(L)$ é uma linguagem regular.

Demonstração.
Seja S uma expressão regular que especifica a linguagem L. Pelo Lema 4.3.13, p. 105, $\mathcal{L}(h(S)) = h\big(\mathcal{L}(S)\big)$. Logo, $h(S)$ é uma expressão regular que especifica $h(L)$, pelo que $h(L)$ é uma linguagem regular.

A imagem inversa de uma linguagem regular por meio de um homomorfismo é ainda uma linguagem regular. Assim, ***a classe das linguagens regulares é fechada para as imagens inversas dos homomorfismos***.

Teorema 4.3.15

Sejam L uma linguagem regular sobre um alfabeto Γ e $h\colon \Sigma^\star \to \Gamma^\star$ um homomorfismo. Então $h^{-1}(L)$ é uma linguagem regular.

Demonstração.
Uma vez que L é uma linguagem regular, existe um AFD $A = (Q, \Gamma, \delta_A, s, F)$ reconhecedor de L. Vamos construir um autómato B que para qualquer palavra $w \in \Sigma^\star$ simula o funcionamento do autómato A com a palavra $h(w)$.

Seja $B = (Q, \Sigma, \delta_B, s, F)$, onde $\delta_B(q, a) = {\delta^\star}_A(q, h(a))$ para $q \in Q$ e $a \in \Sigma$. Por indução, prova-se que ${\delta^\star}_B(s, w) = {\delta^\star}_A(s, h(w))$, para qualquer palavra $w \in \Sigma^\star$.

Assim, uma palavra w é reconhecida pelo autómato B se e só se $h(w)$ é reconhecida pelo autómato A. Logo, a linguagem reconhecida pelo autómato B é a linguagem regular $h^{-1}(L)$.

4.3.4 Substituições regulares

A utilização do teorema da substituição, a seguir enunciado, permite simplificar, em muitos casos, a verificação de algumas propriedades das linguagens regulares.

Teorema 4.3.16

Se s é uma substituição de símbolos por linguagens regulares e L é uma linguagem regular então $s(L)$ é uma linguagem regular.

Demonstração.
Seja R uma expressão regular que especifica a linguagem L. Para cada $a \in \Sigma$, seja R_a uma expressão regular que especifica a linguagem $s(a) = L_a$. Mostra-se, por indução sobre a definição de expressão regular, que a expressão $s(R)$, obtida de R por substituição de cada símbolo a por R_a, é uma expressão regular que especifica a linguagem $s(L)$.

Concluímos pois que ***a classe das linguagens regulares é fechada para substituições regulares***.

Exemplo 4.3.17

Sejam L_1 e L_2 linguagens regulares sobre um alfabeto Σ. Consideremos uma substituição em que $s(a) = L_1$ e $s(b) = L_2$, onde $a, b \in \Sigma$. Uma vez que $\{ab\}$ é uma linguagem regular, por aplicação do teorema da substituição concluímos que $L_1 L_2$ é uma linguagem regular.

Sejam L_1 e L_2 linguagens regulares sobre um alfabeto Σ. Consideremos uma substituição em que $s(a) = L_1$ e $s(b) = L_2$, onde $a, b \in \Sigma$. Uma vez que $\{a, b\}$ é uma linguagem regular, por aplicação do teorema da substituição concluímos que $L_1 \cup L_2$ é uma linguagem regular.

Seja L uma linguagem regular sobre um alfabeto Σ. Consideremos uma substituição em que $s(a) = L$, onde $a \in \Sigma$. Uma vez que a^* denota uma linguagem regular, por aplicação do teorema da substituição concluímos que $L^\star$ é uma linguagem regular.

4.4 O lema da bombagem para linguagens regulares

Vimos que é possível especificar as linguagens regulares usando autómatos finitos e expressões regulares. O problema que abordamos agora é o de decidir se uma linguagem é ou não regular.

Como veremos no Capítulo 7, p. 189, não é possível desenvolver um algoritmo para averiguar se uma qualquer linguagem é ou não regular. Diremos assim que este problema é indecidível.

Claramente, todas as linguagens finitas (com um número finito de palavras) são regulares. Basta escrever uma expressão regular (ou construir um autómato reconhecedor) para cada uma das palavras da linguagem.

No entanto nem todas as linguagens são regulares, sendo $L = \{a^n b^n : \ n \in \mathbb{N}\}$ um exemplo típico.

Intuitivamente, um autómato finito reconhecedor da linguagem L, a existir, teria de ter um número infinito de estados que permitisse "contar" o número de a´s para conseguir confirmar que se seguiam exactamente o mesmo número de b´s.

De facto, vamos admitir que $A = (Q, \{a, b\}, \delta, s, F)$ é um AFD que reconhece a linguagem L. Para $i = 1, 2, \ldots$, consideremos os estados da forma $\delta^\star(s, a^i)$. Uma vez que o conjunto de estados de A é finito, tem de existir um estado q tal que $\delta(s, a^n) = q = \delta(s, a^m)$, com $m \neq n$. Para além disso, $\delta^\star(q, b^n) \in F$, uma vez que o autómato reconhece a palavra $a^n b^n \in L$. Sendo assim, também $\delta^\star(s, a^m b^n) = \delta^\star\big(\delta^\star(s, a^m), b^n\big) = \delta^\star(q, b^n) \in F$, ou seja, o autómato A reconhece a palavra $a^m b^n$ com $m \neq n$, a qual não pertence a L.

Chegamos a uma contradição com a premissa de que A é um autómato reconhecedor da linguagem L, o que nos permite concluir que esta linguagem não é regular.

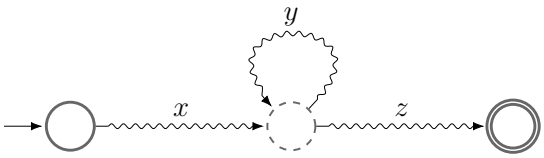
Podemos aplicar a ideia de que os autómatos finitos não conseguem contar para provar que a linguagem $L_1 = \{w \in \{0,1\}^\star : \ \#_0(w) = \#_1(w)\}$ também não é regular.

No entanto, é necessário ter algum cuidado na utilização informal desta ideia. Por exemplo, a linguagem $L_2 = \{w \in \{0,1\}^\star : \ \#_{01}(w) = \#_{10}(w)\}$, aparentemente semelhante a L_1, é regular.

Vamos sistematizar e generalizar a ideia utilizada para provar que a linguagem $L = \{a^n b^n : \ n \in \mathbb{N}\}$ não é regular, apresentando uma condição necessária para que uma linguagem seja regular.

Este resultado, conhecido por lema da bombagem, indica que qualquer palavra suficientemente longa de uma linguagem regular pode ser particionada em três partes (esquerda, central e direita) que satisfazem a seguinte propriedade: todas as palavras obtidas por substituição da parte central por repetições de si mesma, tantas vezes quanto se queira, continuam a fazer parte da linguagem.

Dizemos, por isso, que a parte central é *bombeada* conforme ilustrado em seguida:



Lema 4.4.1 Lema da Bombagem para Linguagens Regulares.

Seja L uma linguagem regular. Então existe um inteiro positivo n tal que, para toda a palavra $w \in L$ com $|w| \geq n$, existem $x, y, z \in \Sigma^\star$ tais que:

1. $w = xyz$;
2. $y \neq \varepsilon$;
3. $|xy| \leq n$;
4. $\forall k \geq 0,\ xy^k z \in L$.

Demonstração.

Sendo L uma linguagem regular, existe um AFD $A = (Q, \Sigma, \delta, s, F)$ que reconhece L. Seja n o número de estados do autómato A e seja $w = a_1 a_2 \cdots a_m \in L$ uma qualquer palavra de tamanho $m = |w| \geq n$. O reconhecimento de w corresponde a um caminho de $m + 1 > n$ estados da forma:

$$s \to \delta^\star(s, a_1) \to \delta^\star(s, a_1 a_2) \to \cdots \to \delta^\star(s, a_1 a_2 \cdots a_m) \in F.$$

Pelo Princípio da Gaiola dos Pombos, entre os $n+1$ primeiros estados há pelo menos dois que são iguais. Sejam então $0 \leq i < j \leq n$ tais que $\delta^\star(s, a_1 a_2 \cdots a_i) = \delta^\star(s, a_1 a_2 \cdots a_j)$.

Fixemos a partição da palavra w nas sub-palavras $x = a_1 \cdots a_i$, $y = a_{i+1} \cdots a_j$ e $z = a_{j+1} \cdots a_m$. Notamos que $y \neq \varepsilon$, uma vez que $i \neq j$. Para além disso, $|xy| \leq n$, uma vez que $j \leq n$. Por indução, vamos agora demonstrar que $\forall k \geq 0$, $\delta^\star(s, xy^k) = \delta^\star(s, xy)$.

Caso base. Para $k = 0$, utilizando a definição de x e y, temos que

$$\delta^\star(s, xy^0) = \delta^\star(s, x) = \delta^\star(s, a_1 a_2 \cdots a_i) = \delta^\star(s, a_1 a_2 \cdots a_j) = \delta^\star(s, xy).$$

Passo indutivo. Para $k > 0$, consideremos a hipótese de indução $\delta^\star(s, xy^k) = \delta^\star(s, xy)$. Temos que

$$\begin{aligned}\delta^\star(s, xy^{k+1}) &= \delta^\star(s, xy^k y) = \delta^\star\big(\delta^\star(s, xy^k), y\big) \\ &= \delta^\star\big(\delta^\star(s, xy), y\big) = \delta^\star\big(\delta^\star(s, x), y\big) = \delta^\star(s, xy).\end{aligned}$$

Assim, podemos também concluir que $\forall k \geq 0$, $\delta^\star(s, xy^k z) = \delta^\star(s, xyz) \in F$ e portanto $\forall k \geq 0$, $xy^k z \in L$.

Quando analisamos uma linguagem, por vezes acontece começar a suspeitar que esta não é regular: quer porque estamos com dificuldade em conseguir construir um autómato determinista que a reconheça, ou em estabelecer uma expressão regular que a defina, ou ainda porque não estamos a conseguir provar a sua regularidade utilizando operações regulares. Nesse caso, podemos sempre tentar mostrar que a linguagem não satisfaz o lema da bombagem.

Uma prova inequívoca de que uma linguagem não verifica o lema da bombagem, obtida geralmente por redução ao absurdo, permite concluir imediatamente que a mesma não pode ser regular.

Teorema 4.4.2

Se uma linguagem não satisfaz o Lema da Bombagem para linguagens regulares então não é uma linguagem regular.

Demonstração.

É consequência imediata do lema da bombagem.

Exemplo 4.4.3

Admitamos que a linguagem $L = \{a^m b^m : \ m \in \mathbb{N}_0\}$ é uma linguagem regular. Então existe $n \in \mathbb{N}$, tal que todas as palavras de L de tamanho não inferior a n satisfazem as condições do lema da bombagem.

Consideremos assim a palavra $w = a^n b^n \in L$.

Uma vez que $|w| \geq n$, existem palavras x, y e z, com $w = xyz$, $|y| \neq 0$, $|xy| \leq n$, tais que $xy^k z \in L$, qualquer que seja $k \geq 0$.

Para além disso, uma vez que $|xy| \leq n$, tanto x como y são necessariamente sequências de a´s, isto é,

$$x = a^p \quad , \quad y = a^q \quad , \quad z = a^r b^n$$

com $p + q + r = n$ e, uma vez que $|y| \neq 0$, $q > 0$.

Ora, pelo Lema da Bombagem, também $xy^0 z \in L$. Mas $xy^0 z = a^{p+r} b^n$, com $p + r \neq n$, pelo que $xy^0 z \notin L$. Chegamos assim a uma contradição, pelo que L não pode ser uma linguagem regular.

Exemplo 4.4.4

Admitamos que $L = \{1^{m^2} : \ m \in \mathbb{N}_0\}$ é uma linguagem regular. Então existe $n \in \mathbb{N}$, tal que todas as palavras de L de tamanho não inferior a n satisfazem as condições do lema da bombagem.

Consideremos a palavra $w = 1^{n^2} \in L$.

Uma vez que $|w| \geq n$, existem palavras x, y e z, com $w = xyz$, $|y| \neq 0$, $|xy| \leq n$, tais que $xy^k z \in L$, qualquer que seja $k \geq 0$.

Existem portanto $p \geq 0$ e $q > 0$ tais que $x = 1^p$, $y = 1^q$ e $z = 1^{n^2-p-q}$.

Pelo lema da bombagem, com $k = 2$, concluímos que

$$xy^2 z = 1^{p+2q+n^2-p-q} = 1^{n^2+q} \in L.$$

Para além disso, $|xy| = p + q \leq n$, donde $q \leq n$. Mas então $n^2 < n^2 + q \leq n^2 + n < n^2 + 2n + 1 = (n + 1)^2$. Logo, $n^2 + q$ não pode ser um quadrado perfeito, pelo que $xy^2 z \notin L$.

Chegamos assim a uma contradição, pelo que L não pode ser uma linguagem regular.

4.5 O teorema de Myhill-Nerode

Vamos agora apresentar uma caracterização teórica da classe das linguagens regulares por intermédio de relações de equivalência sobre o conjunto das palavras, bem como uma sua ligação, à partida não muito evidente, com os estados dos autómatos finitos deterministas mínimos. As origens desta abordagem surgem no final da década de 1950,com os trabalhos independentes desenvolvidos por John Myhill (Reino Unido, 1923 -1987)[17] e Anil Nerode (Estados Unidos da América, 1932 -)

[18].

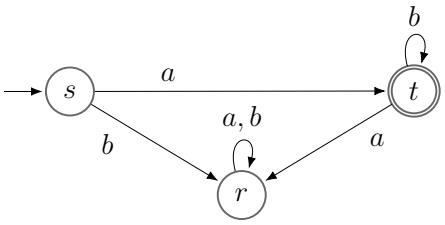
Sejam $L \subseteq \Sigma^\star$ uma linguagem regular e $A = (Q, \Sigma, \delta, s, F)$ um AFD acessível, completo e reconhecedor de L. O autómato induz uma relação de equivalência em $\Sigma^\star$ definida por

$$x \sim_A y \text{ se e só se } \delta^\star(s, x) = \delta^\star(s, y). \tag{4.5.1}$$

Podemos verificar que as classes de equivalência da relação $\sim_A$ são tantas quantos os estados do autómato, cada uma delas idêntica à linguagem esquerda de um dos estados do autómato.

Exemplo 4.5.1

Seja L a linguagem associada à expressão regular ab^*. O AFD A a seguir ilustrado reconhece L.



Verifica-se facilmente que as classes de equivalência da relação $\sim_A$ são:

- $[\varepsilon] = \varepsilon$;
- $[a] = ab^*$;
- $[b] = (b + ab^*a)(a + b)^*$.

Para além de ser uma relação de equivalência, a relação $\sim_A$ é ainda uma relação de Myhill-Nerode, o que significa que satisfaz as propriedades adicionais de ser uma congruência à direita, que refina L e que tem índice finito. Explicitamos o significado preciso destas propriedades na definição que se segue.

Definição 4.5.2

Uma relação de equivalência $\sim$ em $\Sigma^\star$ é uma *relação de Myhill-Nerode* para uma linguagem $L \subseteq \Sigma^\star$ se

(1) é uma *congruência à direita*, isto é, para quaisquer $x, y \in \Sigma^\star$ e $a \in \Sigma$,

$$\text{se } x \sim y \text{ então } xa \sim ya;$$

(2) *refina* L, isto é, para quaisquer $x, y \in \Sigma^\star$,

$$\text{se } x \sim y \text{ então } x \in L \Leftrightarrow y \in L;$$

(3) tem *índice finito*, isto é, tem um número finito de classes de equivalência distintas.

Seja $\sim$ uma *relação de Myhill-Nerode* para uma linguagem $L \subseteq \Sigma^\star$. A cada palavra $w \in \Sigma^\star$ corresponde uma classe de equivalência $[w] = \{x \in \Sigma^\star : \ x \sim w\}$. Embora exista um número infinito de palavras, existe um número finito de classes de equivalência (pela terceira condição da Definição 4.5.2, p. 112). Vamos associar à relação $\sim$ um autómato finito determinista $A_\sim$ especificado na definição seguinte.

Definição 4.5.3

O AFD associado a uma relação de Myhill-Nerode $\sim$ para uma linguagem $L \subseteq \Sigma^\star$ é

$$A_\sim = (Q, \Sigma, \delta, s, F)\,,$$

definido por:

- $Q = \{[w] : \ w \in \Sigma^\star\}$;
- $s = [\varepsilon]$;
- $F = \{[w] : \ w \in L\}$;
- para $w \in \Sigma^\star$ e $a \in \Sigma$, $\delta([w], a) = [wa]$.

A função de transição δ está bem definida uma vez que a relação $\sim$ é uma congruência à direita (pela primeira propriedade da Definição 4.5.2, p. 112). Para além disso, atendendo à definição de F, verifica-se que

$$w \in L \text{ se e só se } [w] \in F. \tag{4.5.2}$$

Finalmente, prova-se por indução que

$$\delta^\star([x], y) = [xy], \tag{4.5.3}$$

para quaisquer palavras $x, y \in \Sigma^\star$.

Vamos demonstrar que as linguagens que admitem uma relação de Myhill-Nerode $\sim$ são aquelas para as quais existe um AFD $A_\sim$, pelo que são linguagens regulares.

Teorema 4.5.4

Se $\sim$ é uma relação de Myhill-Nerode *para uma linguagem $L \subseteq \Sigma^\star$ então a linguagem reconhecida pelo* AFD *$A_\sim$ é idêntica a L.*

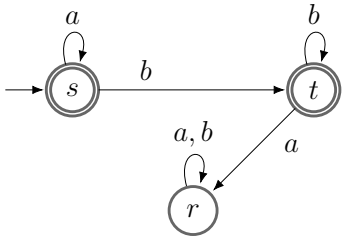
Demonstração.
Seja $w \in \Sigma^\star$. Pela definição de linguagem reconhecida por um AFD, $x \in A_\sim$ se e só se $\delta^\star([\varepsilon], w) \in F$. Pela propriedade (4.5.3), $\delta^\star([\varepsilon], w) \in F$ se e só se $[w] \in F$. Finalmente, pela propriedade (4.5.2), $[w] \in F$ se e só se $w \in L$.

Exemplo 4.5.5

Consideremos a relação de equivalência $\sim$ em $\{a, b\}^\star$ definida pelas seguintes classes de equivalência:

- $[\varepsilon] = a^*$;
- $[b] = a^*bb^*$;
- $[ba] = a^*bb^*a(a + b)^*$.

Deixamos como exercício verificar que $\sim$ é uma relação de Myhill-Nerode para a linguagem associada à expressão regular a^*b^*. O AFD $A_\sim$ obtido a partir da relação $\sim$ é o seguinte:



Descrevemos agora duas construções que são essencialmente inversas uma da outra: a primeira transforma um autómato A reconhecedor de uma linguagem L numa relação $\sim_A$ de Myhill-Nerode para a linguagem L; a segunda transforma uma relação de Myhill-Nerode $\sim$ para uma linguagem L num autómato $A_\sim$ que reconhece L.

Lema 4.5.6

Seja $A_\sim$ o AFD *associado a uma relação de Myhill-Nerode $\sim$ para uma linguagem L. Seja $\sim_{A_\sim}$ a relação de Myhill-Nerode obtida a partir do* AFD *$A_\sim$, usando (4.5.1). Então as relações $\sim_{A_\sim}$ e $\sim$ são iguais.*

Demonstração.

Seja $A_\sim = (Q, \Sigma, \delta, s, F)$ o AFD que se obtém aplicando a primeira construção. Para $x, y \in \Sigma^\star$, temos que: $x \sim_{A_\sim} y$ se e só se $\delta^\star(s, x) = \delta^\star(s, y)$ se e só se $\delta^\star([\varepsilon], x) = \delta^\star([\varepsilon], y)$ se e só se $[x] = [y]$ se e só se $x \sim y$.

Lema 4.5.7

Seja $\sim_A$ a relação de Myhill-Nerode obtida a partir de um AFD *acessível, A, usando (4.5.1). Seja $A_{\sim_A}$ o autómato associado à relação $\sim_A$. Então os autómatos $A_{\sim_A}$ e A são isomorfos.*

Demonstração.

Sejam $A = (Q, \Sigma, \delta, s, F)$ e $A_{\sim_A} = (Q', \Sigma, \delta', s', F')$, onde:

- $[x]_{\sim_A} = \{y : \ y \sim_A x\} = \{y : \ \delta^\star(s, y) = \delta^\star(s, x)\}$;
- $Q' = \{[x]_{\sim_A} : \ x \in \Sigma^\star\}$;
- $s' = [\varepsilon]_{\sim_A}$;
- $F' = \{[x]_{\sim_A} : \ x \in F\}$;
- $\delta'([x]_{\sim_A}, a = [xa]_{\sim_A}$ para $x \in \Sigma^\star$ e $a \in \Sigma$.

Vamos mostrar que os autómatos $A_{\sim_A}$ e A são isomorfos por intermédio do isomorfismo $h \colon Q' \to Q$ definido por $h([x]_{\sim_A}) = \delta^\star(s, x)$, para $x \in \Sigma^\star$.

Pela definição de $\sim_A$, para $x, y \in \Sigma^\star$, temos que $[x]_{\sim_A} = [y]_{\sim_A}$ se e só se $\delta^\star(s, x) = \delta^\star(s, y)$. Assim a função h está bem definida e é injectiva. Uma vez que o autómato não tem estados inacessíveis, a função h é também sobrejectiva.

Temos ainda que:

- $h(s') = h([\varepsilon]_{\sim_A}) = \delta^\star(s, \varepsilon) = s$;
- para $x \in \Sigma^\star$ e para $a \in \Sigma$, $h(\delta'([x]_{\sim_A}, a)) = h([xa]_{\sim_A}) = \delta^\star(s, xa) = \delta(\delta^\star(s, x)) = \delta(h([x]_{\sim_A}, a)$;
- para $x \in \Sigma^\star$, $[x]_{\sim_A} \in F'$ se e só se $x \in F$ se e só se $\delta^\star(s, x) \in F$ se e só se $h([x]_{\sim_A}) \in F$.

Assim, h é uma bijecção de estados que preserva o estado inicial, as transições e ainda o conjunto de estados de aceitação, ou seja, h é um isomorfismo de autómatos.

Do exposto acima, concluímos que existe uma correspondência biunívoca entre a classe dos autómatos finitos deterministas acessíveis e completos, reconhecedores de uma linguagem regular L e as relações de Myhill-Nerode para L (a menos de um isomorfismo de autómatos).

Encarando uma relação num conjunto X como um conjunto de pares ordenados em $X \times X$, dizemos que a relação $\sim_1$ é mais fina do que a relação $\sim_2$ sempre que $\sim_1 \subseteq \sim_2$. Nesse caso, dizemos ainda que $\sim_1$ refina $\sim_2$.

Quando restrita a relações de equivalência, $\sim_1$ é mais fina do que $\sim_2$ sempre que $[x]_{\sim_1} \subseteq [x]_{\sim_2}$, para qualquer $x \in X$. O refinamento entre relações de equivalência define uma ordem parcial (é reflexiva, transitiva e anti-simétrica). Para cada conjunto X, a relação identidade $\{(x, x) : \ x \in X\}$ é a relação de equivalência mais fina de todas e a relação universal $\{(x, y) : \ x, y \in X\}$ é a menos fina de todas.

Seja $L \subseteq \Sigma^\star$ uma linguagem e consideremos a relação em $\Sigma^\star$ definida por

$$x \sim_L y \text{ se e só se } \forall z \in \Sigma^\star, xz \in L \Leftrightarrow yz \in L \tag{4.5.4}$$

A relação $\sim_L$ é de equivalência entre palavras. Duas palavras x e y são equivalentes quando se "comportam" da mesma forma ao serem concatenadas à direita com uma mesma palavra z arbitrária: as palavras xz e yz pertencem ambas à linguagem L ou nenhuma pertence à linguagem L.

Vamos verificar que a relação $\sim_L$ satisfaz as propriedades (1) e (2) da Definição 4.5.2, p. 112, qualquer que seja a linguagem L (regular ou não), sendo ainda a menos fina das relações de equivalência que satisfazem aquelas duas propriedades.

Para além disso, se a linguagem L for regular então $\sim_L$ tem índice finito e é portanto uma relação de Myhill-Nerode para L.Nesse caso, pelo facto de ser a relação menos fina de Myhill-Nerode para L, corresponde também ao AFD mínimo que aceita a linguagem L.

Lema 4.5.8

Seja L uma linguagem sobre um alfabeto Σ. A relação $\sim_L$ definida por (4.5.4) é uma congruência à direita que refina L.

Demonstração.

Para mostrar que $\sim_L$ é uma congruência à direita basta substituir z por aw, com $a \in \Sigma$ e $w \in \Sigma^\star$ na definição de $\sim_L$.

Para mostrar que $\sim_L$ refina L basta considerar $z = \varepsilon$ na definição de $\sim_L$.

Lema 4.5.9

Seja L uma linguagem sobre um alfabeto Σ. A relação $\sim_L$ definida por (4.5.4) é a menos fina de entre todas as congruências à direita que refinam L.

Demonstração.

Vamos verificar que qualquer relação $\sim$ que satisfaça as propriedades (1) e (2) da Definição 4.5.2, p. 112 é uma relação que refina $\sim_L$.

Sejam então x e y palavras tais que $x \sim y$. Usando a propriedade (1) e indução sobre $|z|$ mostramos que para qualquer palavra $z \in \Sigma^\star$, $xz \sim yz$. Assim, pela propriedade (2), para qualquer palavra z, $xz \in L$ se e só se $yz \in L$, ou seja, $x \sim_L y$.

Teorema 4.5.10

Seja L uma linguagem sobre um alfabeto Σ. As seguintes afirmações são equivalentes:

(a) L é uma linguagem regular;

(b) Existe uma relação de Myhill-Nerode para L;

(c) A relação $\sim_L$ tem índice finito.

Demonstração.

(a) $\implies$ *(b).* Dado um AFD A, reconhecedor de L, a transformação $A \mapsto \sim_A$ produz uma relação de Myhill-Nerode para L.

(b) $\implies$ *(c).* Pelos Lemas 4.5.8, p. 115 e 4.5.9, p. 116 qualquer relação de Myhill-Nerode tem índice finito e refina $\sim_L$. Logo, $\sim_L$ também tem índice finito.

(c) $\implies$ *(a).* Se $\sim_L$ tem índice finito então é uma relação de Myhill-Nerode e a construção $\sim \mapsto A_\sim$ produz um AFD que reconhece L.

A relação de Myhill-Nerode menos fina de todas, $\sim_L$, corresponde ao AFD que reconhece L com o menor número de estados entre todos os autómatos finitos deterministas (acessíveis e completos) que reconhecem L. Como vimos na Secção 3.2, p. 73, o AFD quociente $A_\equiv$ obtido com o algoritmo de identificação de estados é um autómato acessível, completo e reduzido pelo que a relação de indistinguibilidade de estados no autómato quociente é a identidade. Assim, a relação de Myhill-Nerode $\equiv_A$ associada ao autómato quociente é idêntica à relação $\sim_L$. Deixamos como exercício a demonstração desta última afirmação.

Para além de providenciar uma caracterização teórica da classe das linguagens regulares, o teorema de Myhill-Nerode pode ser aplicado ao problema de decidir se uma linguagem L é ou não regular, bastando para isso determinar se o número

de classes da relação $\sim_L$ é finito ou infinito.

Exemplo 4.5.11

Consideremos a linguagem $L = \{11w : \ w \in \{0,1\}^\star\}$.

Temos que $110 \ \sim_L \ 1110010$ uma vez que $110z \in L$ e $1110010z \in L$ para qualquer palavra $z \in \{0,1\}^\star$.

Assim, uma das classes de equivalência é constituída pelas palavras binárias que começam com 11:

$$[11]_{\sim_L} = [110]_{\sim_L} = [1110010]_{\sim_L} = L.$$

Notamos que $11 \not\sim_L 10$, uma vez que existe uma palavra z, por exemplo $z = \varepsilon$ ou $z = 1$, tal que $11z \in L$ enquanto $10z \notin L$. Assim, $[11]_{\sim_L} \neq [10]_{\sim_L}$.

Para além disso, $[0]_{\sim_L} = [01]_{\sim_L} = [10]_{\sim_L}$.

Temos ainda que $[1]_{\sim_L} \neq [0]_{\sim_L}$, uma vez que para $z = 1$ se verifica que $1z \in L$ enquanto $0z \notin L$.

De forma análoga concluímos que $[1]_{\sim_L} \neq [11]_{\sim_L}$, uma vez que para $z = 0$ se verifica que $1z \notin L$ enquanto $11z \in L$.

Finalmente, podemos verificar facilmente que $[\varepsilon]_{\sim_L} = \{\varepsilon\}$.

Assim, a relação $\sim_L$ tem apenas quatro classes de equivalência distintas: $[\varepsilon]_{\sim_L}$, $[0]_{\sim_L}$, $[1]_{\sim_L}$ e $[11]_{\sim_L}$.

Uma vez que a relação $\sim_L$ tem índice finito, a linguagem L é regular.

Exemplo 4.5.12

Consideremos a linguagem $L = \{a^n b^n : \ n \geq 0\}$.

Se $m \neq n$ então $a^m \not\sim a^n$, uma vez que $a^m b^m \in L$ mas $a^n b^m \notin L$. Assim, as classes distintas da relação $\sim_L$ são $[a^m]$, com $m \geq 0$.

Uma vez que existe um número infinito de classes distintas, o teorema de Myhill-Nerode garante que L não é regular.

4.6 Exercícios

1. Construa expressões regulares para as linguagens referidas nos exercícios 2.6.2, p. 59, 2.6.3, p. 59 e 2.6.10, p. 60.
2. Considere, como ponto de partida, que uma linguagem regular é uma linguagem associada a uma expressão regular. Diga, justificando, qual o valor

lógico de cada uma das seguintes afirmações:

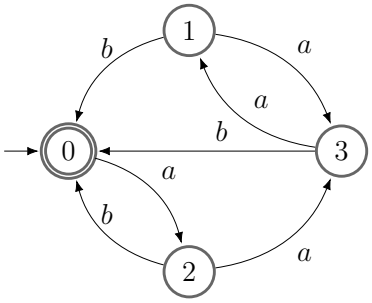
(a) $\forall n \in \mathbb{N}$, se $\forall i \in \{1, \ldots, n\}$, L_i é uma linguagem regular então $L = \bigcup_{i=1}^{n} L_i$ é uma linguagem regular.

(b) $\forall n \in \mathbb{N}$, se $\forall i \in \{1, \ldots, n\}$, L_i é uma linguagem regular então $L = \bigcap_{i=1}^{n} L_i$ é uma linguagem regular.

(c) Se $\forall n \in \mathbb{N}$, L_n é uma linguagem regular então $L = \bigcup_{n=1}^{\infty} L_n$ é uma linguagem regular.

(d) Se $\forall n \in \mathbb{N}$, L_n é uma linguagem regular então $L = \bigcap_{n=1}^{\infty} L_n$ é uma linguagem regular.

3. Considere o AFD A definido pelo seguinte diagrama de transições:



(a) Caracterize as palavras da linguagem $\mathcal{L}(A)$.

(b) Usando o método de identificação de estados indistinguíveis, determine se o autómato A é ou não mínimo. Indique quantas iterações realizou e, para cada par de estados distinguíveis, apresente uma palavra que os distinga.

(c) Não sendo mínimo, apresente o diagrama de transições do autómato mínimo reconhecedor da linguagem $\mathcal{L}(()A)$.

(d) Indique uma expressão regular R tal que $\mathcal{L}(R) = \mathcal{L}(A)$.

(e) Determine uma expressão regular S tal que $\mathcal{L}(S) = \mathcal{L}(A)^{-1}$.

4. Demonstre o Lema 4.3.13, p. 105.

5. Sejam $h\colon \Sigma^\star \to \Sigma^\star$ um homomorfismo de palavras e L uma linguagem sobre o alfabeto Σ. Diga, justificando, se as seguintes afirmações são verdadeiras ou falsas:

(a) Se L não é regular então $h(L)$ não é regular.

(b) Se $h(L)$ não é regular então L não é regular.

(c) Se L não é regular então $h^{-1}(L)$ não é regular.

(d) Se $h^{-1}(L)$ não é regular então L não é regular.

6. Considere os alfabetos $\Sigma = \{a, b\}$, $\Gamma = \{0, 1\}$ e $h\colon \Sigma \to \Gamma$ o homomorfismo definido por $h(a) = 00$, $h(b) = 1$. Considere a expressão regular $S = a^*b + ba^*$.

(a) Especifique uma expressão regular para a linguagem $h(\mathcal{L}(S))$.

(b) Defina em compreensão as linguagens $\mathcal{L}(S)$ e $h(\mathcal{L}(S))$.

(c) Determine expressões regulares para as linguagens $L_1 = h^{-1}(\{0011\}^\star)$ e $L_2 = h^{-1}(\{01\}\{01\}^\star)$.

7. Sejam R e S expressões regulares tais que $\mathcal{L}(R) \subseteq \mathcal{L}(S)$. Mostre que $R+S = S$.

8. Sejam A, B e X expressões regulares tais que $\mathcal{L}(A + XB) \subseteq \mathcal{L}(X)$. Mostre que $\mathcal{L}(AB^*) \subseteq \mathcal{L}(X)$

9. Sejam R, S e T expressões regulares quaisquer. Prove que:

(a) $R^*R^* = R^*$;

(b) $R^*S + S = R^*S$;

(c) $R + SS^*R = S^*R$;

(d) $R^*(\varepsilon + R) = (\varepsilon + R)R^* = R^*$;

(e) $(R + S)^* = (R^*S^*)^*$;

(f) $(\varepsilon + R)^* = R^*$;

(g) $(\varepsilon + R)^+ = R^*$;

(h) $R(SR)^* = (RS)^*R$;

(i) $(R + S)^* = (R^*S)^*R^* = (S^*R)^*S^*$.

10. Seja L uma linguagem regular, definida sobre um alfabeto Σ com pelo menos dois símbolos.

(a) Mostre que $L_1 = \{w :\ ww \in L\}$ é regular.

(b) Mostre que $L_2 = \{ww :\ w \in L\}$ pode não ser regular.

11. Recordando que uma linguagem é livre de prefixos se os prefixos próprios das palavras da linguagem não são palavras da linguagem, mostre que:

(a) Uma linguagem regular, não vazia, é livre de prefixos se e só se é reconhecível por um AFD com um único estado de aceitação em que as transições a partir do estado de aceitação são para um estado ratoeira.

(b) Se R é uma expressão regular associada a uma linguagem regular livre de prefixos então $\mathcal{L}(R^*)$ é reconhecível por um AFD com um único estado de aceitação.

(c) Seja A um AFD em que o único estado de aceitação é o estado inicial. Então $\mathcal{L}(A) = \mathcal{L}(R^*)$, para alguma expressão regular R associada a

uma linguagem regular livre de prefixos.

(d) Uma linguagem L é reconhecida por um AFD com um único estado de aceitação se e só se $L = \mathcal{L}(SR^*)$, onde S e R são expressões regulares associadas a linguagens regulares livres de prefixos.

12. Mostre que qualquer linguagem regular se pode escrever como a união de um número finito de linguagens regulares, cada uma delas aceite por um AFD com um único estado de aceitação.

13. Mostre que o *quociente de duas linguagens* regulares L_1 e L_2 é uma linguagem regular, isto é:

$$L_1 \,/\, L_2 = \{x \in \Sigma^\star : \ \text{existe } y \in L_2 \text{ tal que } xy \in L_1\}.$$

14. Mostre que qualquer linguagem finita satisfaz o lema da bombagem para linguagens regulares.

15. Averígue se as seguintes linguagens são ou não regulares:

(a) $L = \{0^n 10^n : \ n \geq 1\}$;

Sugestão. Aplique o lema da bombagem com $w = 0^n 10^n$: verifique que as partições $w = xyz$ são todas da forma $x = a^p$, $y = a^q$, com $q \geq 1$ e $z = a^{n-p-q}10^n$; Faça $k = 0$ para obter uma contradição.

(b) $L = \{0^n : \ n$ é um número primo$\}$;

Sugestão. Aplique o lema da bombagem com $w = 0^p$ para algum primo $p > n$: verifique que as partições $w = xyz$ são todas da forma $x = 0^q$, $y = 0^r$, com $r \geq 1$ e $z = 0^{p-q-r}$; Faça $k = p + 1$ e conclua que $p(r + 1)$ não é primo.

(c) $L = \{0^{n!} : \ n \geq 0\}$;

Sugestão. Aplique o lema da bombagem com $w = 0^{m!}$ com $m > \max\{2, n\}$: verifique que as partições $w = xyz$ são todas da forma $x = 0^p$, $y = 0^q$, com $1 \leq q \leq m$ e $z = 0^{m!-p-q}$; Faça $k = 2$ e conclua que $m! < m! + q < (m + 1)!$.

(d) $L = \{a^m b^n c^{m+n} : \ m, n \geq 0\}$;

Sugestão. Aplique o lema da bombagem com $w = b^n c^n$.

(e) $L = \{w \in \{a, b\}^\star : \ w = w^{-1}\}$;

Sugestão. Aplique o lema da bombagem com $w = a^n b a^n$.

(f) $L = \{w \in \{a, b\}^\star : \ w \neq w^{-1}\}$;

(g) $L = \{ww : \ w \in \{a, b\}^\star\}$;

Sugestão. Aplique o lema da bombagem com $w = a^n b a^n b$.

(h) $L = \{ww^{-1} : \ w \in \{a, b\}^\star\}$;

Sugestão. Aplique o lema da bombagem com $w = a^n bba^n$.

(i) $L = \{w \in \{a, b\}^\star : \ \#_a(w) = \#_b(w)\}$;

Sugestão. Aplique o lema da bombagem com $w = a^n bba^n$.

(j) $L = \{a^n b^m : \ 0 \leq n < m\}$;

Sugestão. Aplique o lema da bombagem com $w = a^n b^{n+1}$ e faça $k = 2$.

(k) $L = \{a^m b^n : \ 0 \leq n < m\}$;

Sugestão. Aplique o lema da bombagem com $w = a^{n+1} b^n$ e faça $k = 0$.

16. Mostre que a relação $\sim_A$ definida pela condição (4.5.1) é uma relação de Myhill-Nerode.

17. Demonstre a propriedade (4.5.3).

18. Seja A o autómato mínimo (a menos de um isomorfismo) para uma linguagem regular L. Mostre que a relação de Myhill-Nerode $\equiv_A$ associada a este autómato é idêntica à relação $\equiv_L$.

19. Aplique o Teorema de Myhill-Nerode para averiguar se as linguagens seguintes são regulares. Em seguida, partindo das classes de equivalência obtidas, construa o autómato determinista mínimo correspondente.

(a) $L = \{aw : \ w \in \{a, b\}^\star\}$;

(b) $L = \{w \in \{0, 1\}^\star : \ \text{a soma dos dígitos de } w \text{ é divisível por } 6\}$;

(c) $L = \{w \in \{a, b\}^\star : \ w = w^{-1}\}$.