1 As Bases

1.4 Exercícios

1.4.1. Caracterize as linguagens $L_1 \cup \overline{L_2}$, $\overline{L_1} \cap L_2$, $L_1\overline{L_2}$ e $\overline{L_2}L_1$, onde L_1 e L_2 são as linguagens definidas no Exemplo 1.2.9.

Solução. $L_1 \cup \overline{L_2}$ é o conjunto das palavras que não começam com ab ou terminam em b.

$\overline{L_1} \cap L_2$ é o conjunto das palavras que começam com ab e terminam em a.

$L_1\overline{L_2}$ é o conjunto das palavras que terminam em b ou terminam em ba ou contêm bb ou contêm baa.

$\overline{L_2}L_1$ é o conjunto das palavras que terminam em b.

1.4.2. É verdade que o fecho de Kleene de uma qualquer linguagem é sempre uma linguagem infinita?

Resolução. Se a linguagem tem pelo menos uma palavra diferente da palavra vazia então o seu fecho de Kleene é uma linguagem infinita. Assim, as duas únicas linguagens cujos fechos de Kleene são linguagens finitas são $L_1 = \emptyset$ (sendo $L_1^\star = \{\varepsilon\}$) e $L_2 = \{\varepsilon\}$ (sendo $L_2^\star = \{\varepsilon\}$).

1.4.3. Seja $\Sigma = \{a, b\}$.

(a) Quantas palavras em $\Sigma^\star$ têm tamanho 0? E tamanho 1? E tamanho 2?

Resolução. Existe uma única palavra de tamanho 0: a palavra vazia. Existem duas palavras de tamanho 1: as palavras a e b. Existem 4 palavras de tamanho 2: as palavras aa, ab, ba e bb.

(b) Para $n \in \mathbb{N}_0$, estabeleça uma fórmula para o número de palavras em $\Sigma^\star$ de tamanho n e demonstre-a.

Resolução. Para $n \in \mathbb{N}_0$, o número de palavras de tamanho n é dado por

$|\Sigma^n| = |\Sigma|^n = 2^n$.

Esta propriedade é válida para $n = 0$ pois a única palavra de tamanho 0 é ε. Admita-se que o número de palavras de tamanho $n-1$ é 2^{n-1}. Cada palavra de tamanho n é a concatenação de um prefixo de tamanho $n-1$ com um sufixo de tamanho 1. Como existem 2 palavras de tamanho 1, existem $2^{n-1} \times 2 = 2^n$ palavras de tamanho n.

(c) O conjunto $\Sigma^\star$ é enumerável?

Resolução. Sim, $\Sigma^\star = \bigcup_{n=0}^{\infty} \Sigma^n$ é uma união enumerável de conjuntos finitos, pelo que é enumerável.

1.4.4. Para $\Sigma = \{0, 1\}$ e $n \in \mathbb{N}_0$, seja $L_n = \{w \in \Sigma^\star : \ |w| \leq n\}$. Qual é o cardinal de L_n?

Solução. $|L_n| = 2^{n+1} - 1$.

1.4.5. Seja Σ um alfabeto de tamanho m. Para $n \in \mathbb{N}_0$, estabeleça uma fórmula para o número de palavras em $\Sigma^\star$ com tamanho n e demonstre-a.

Solução. $|\Sigma^n| = |\Sigma|^n = m^n$. (Ver a resolução do exercício 1.4.3.)

1.4.6. Seja L uma linguagem. Mostre, usando as definições de $L^\star$ e L^+, que:

(a) $L^+ = LL^\star$;

Resolução. $LL^\star = L \cup_{n=0}^{\infty} L^n = \cup_{n=0}^{\infty} LL^n = \cup_{n=0}^{\infty} L^{n+1} = \cup_{n=1}^{\infty} L^n = L^+$.

(b) $L^\star \setminus L^+ = \{\varepsilon\}$.

Resolução. Seja $w \in L^\star$. Então existe $n \in \mathbb{N}_0$, tal que $w \in L^n$.

Se $n = 0$ então $w \in L^\star$ e $w \notin L^+$. Logo, $w \in L^\star \setminus L^+$. Por outro lado, se $w \in L^0 = \{\varepsilon\}$ então $w = \varepsilon$. Logo, $\{\varepsilon\} \subseteq L^\star \setminus L^+$.

Se $n \geq 1$ então $w \in L^\star$ e $w \in L^+$. Logo, $w \notin L^\star \setminus L^+$.

1.4.7. Diga, justificando, se as seguintes afirmações são verdadeiras ou falsas:

(a) Estima-se que na Via Láctea existam cerca de $10^{12} = 100000000000$ estrelas. Embora muitas, as estrelas na nossa galáxia não são tantas quantas as linguagens definidas sobre o alfabeto $\Sigma = \{\bigstar\}$ e que são constituídas por palavras de tamanho inferior a 48.

Resolução. Existem exactamente 48 palavras definidas sobre o alfabeto $\{\bigstar\}$ de tamanho inferior a 48: ε (tamanho 0), $\bigstar$ (tamanho 1), $\bigstar\bigstar$ (tamanho 2), ..., $\bigstar\bigstar\cdots\bigstar$ (tamanho 47).

Uma vez que o número de linguagens que se podem formar com estas 48 palavras é dado por $2^{48} = 16^{12} > 10^{12}$, concluímos que a afirmação é verdadeira.

(b) Seja x uma qualquer palavra sobre um alfabeto $\Sigma = \{a, b\}$. Não existem 4 palavras y de tamanho $|y| = 3$ que façam com que xyx^{-1} seja um palíndromo ímpar.

Resolução. Temos que $(xyx^{-1})^{-1} = (x^{-1})^{-1}y^{-1}x^{-1} = xy^{-1}x^{-1}$. Assim, para que xyx^{-1} seja um palíndromo, y tem de ser um palíndromo.

Por outro lado, $|xyx^{-1}| = 2|x| + |y|$. Assim, para que xyx^{-1} seja um palíndromo ímpar, y tem de ser um palíndromo ímpar.

As palavras sobre o alfabeto $\{a, b\}$ com tamanho 3 que são palíndromos ímpares são: aaa, bbb, aba e bab. Concluímos que a afirmação é falsa.

1.4.8. Mostre que o conjunto de todas as linguagens sobre um qualquer alfabeto Σ não é enumerável.

Resolução. Vamos usar o método da diagonalização de Cantor (Georg Cantor, Alemanha, 1845 - 1918), para mostrar, por redução ao absurdo, que o conjunto de todas as linguagens sobre o alfabeto Σ, $\mathcal{L} = \{L : \ L \subseteq \Sigma^\star\}$, não é enumerável.

Se $\mathcal{L}$ é um conjunto enumerável então podemos enumerar todas as linguagens sobre o alfabeto Σ: $L_1, L_2, L_3, \ldots$ (Existe uma função bijectiva de $\mathbb{N}$ em $\mathcal{L}$ que a cada $n \in \mathbb{N}$ faz corresponder uma linguagem L_n.)

Por outro lado, o conjunto de todas as palavras sobre o alfabeto Σ é também enumerável, pelo que podemos enumerar todas as palavras: $w_1, w_2, w_3, \ldots$ (Existe uma função bijectiva de $\mathbb{N}$ em $\Sigma^\star$.)

Consideremos a linguagem $L = \{w_i \in \Sigma^\star : \ w_i \notin L_i, i \in \mathbb{N}\}$. Para cada $i \in \mathbb{N}$, constatamos que a linguagem L é diferente da linguagem L_i, uma vez que a palavra w_i pertence ou a L ou a L_i. Assim, a linguagem L é uma linguagem que não está na enumeração de todas linguagens, o que é absurdo.

1.4.9. Seja Σ um qualquer alfabeto. Diga, justificando, qual é o valor lógico das seguintes afirmações:

(a) $\Sigma^\star$ é um conjunto finito.

Resolução. A afirmação é falsa. O conjunto $\Sigma^\star = \cup_{n\in\mathbb{N}_0}\Sigma^i$ é uma união infinita de conjuntos não vazios e disjuntos, pelo que é sempre infinita.

Por exemplo, para o alfabeto $\Sigma = \{1\}$ temos que $\Sigma^n = \{1^n\}$, para $n \in \mathbb{N}_0$. A função $v : \Sigma^\star \to \mathbb{N}_0$ definida por $v(1^n) = n$ é uma bijecção entre $\Sigma^\star$ e $\mathbb{N}_0$, donde $\Sigma^\star$ tem a mesma cardinalidade que $\mathbb{N}_0$.

(b) Para $x, y \in \Sigma^\star$, se $x = x^{-1}$ e $y = y^{-1}$ então $(xy)^{-1} = yx$.

Resolução. A afirmação é verdadeira. De facto, para quaisquer palavras x, y, $(xy)^{-1} = y^{-1}x^{-1}$. Uma vez que $x = x^{-1}$ e $y = y^{-1}$ então também $(xy)^{-1} = yx$.

(c) Se L é uma qualquer linguagem então $L^3 = \{w^3 : \ w \in L\}$.

Resolução. A afirmação é falsa. Por exemplo, para $L = \{a, b\}$, $aba \in L^3$ e $aba \neq a^3$ e $aba \neq b^3$.

1.4.10. Seja x uma qualquer palavra sobre um alfabeto $\Sigma = \{a, b\}$. Será que é sempre possível definir uma palavra y, de tamanho $|y| \geq 2$, de tal modo que a palavra xyx^{-1} seja um palíndromo ímpar?

Resolução. Temos que $(xyx^{-1})^{-1} = (x^{-1})^{-1}y^{-1}x^{-1} = xy^{-1}x^{-1}$. Assim, para que xyx^{-1} seja um palíndromo, y tem de ser um palíndromo.

Por outro lado, $|xyx^{-1}| = 2|x| + |y|$. Assim, para que xyx^{-1} seja um palíndromo ímpar, y tem de ser um palíndromo ímpar.

Naturalmente, existe uma infinidade de palíndromos ímpares y que se podem definir sobre o alfabeto $\{a, b\}$, com tamanho $|y| \geq 2$, por exemplo, o palíndromo $y = aaa$. Concluímos que a afirmação é verdadeira.

1.4.11. Apresente exemplos de linguagens L e valores de $n \in \mathbb{N}_0$ para os quais se verifique

$$L^n = \{w^n : \ w \in L\}.$$

Resolução. Se $L = \emptyset$ então, por definição, $L^0 = \{\varepsilon\}$ enquanto $\{w^0 : \ w \in \emptyset\} = \emptyset$. No entanto, para $n \in \mathbb{N}$, $L^n = \emptyset = \{w^n : \ w \in \emptyset\}$.

Se $L \neq \emptyset$ então, por definição, $L^0 = \{\varepsilon\} = \{w^0 : \ w \in L\}$.

Se $n = 1$ então $L^1 = L = \{w^1 : \ w \in L\}$.

Se $L = \{\varepsilon\}$ então para $n \in \mathbb{N}_0$, $L^n = \{\varepsilon\} = \{w^n : \ w \in L\}$.

Se uma linguagem possui uma só palavra w então, para $n \in \mathbb{N}_0$, $L^n = \{w^n : \ w \in L\}$.

A propriedade não se verifica para $n \geq 2$ sempre que a linguagem possui pelo menos duas palavras. Por exemplo, para $L = \{a, ab\}$ tem-se que $L^2 = \{aa, aab, aba, abab\}$ enquanto $\{w^2 : \ w \in L\} = \{aa, abab\}$.

1.4.12. Partindo da definição indutiva de potência de uma palavra do Exemplo 1.3.1, mostre que $\forall n \in \mathbb{N}_0, \forall w \in \Sigma^\star, w^{n+1} = w^n w$.

Resolução. Vamos usar indução sobre $n \in \mathbb{N}_0$ para demonstrar a propriedade enunciada.

Caso base. Para $n = 0$, $w^{n+1} = w^1 = w = \varepsilon w = w^0 w = w^n w$.

Indução. Consideremos por hipótese que $w^{n+1} = w^n w$ e vamos provar que

$w^{n+2} = w^{n+1}w$:

$$\begin{aligned} w^{n+2} &= w^{(n+1)+1} \\ &= ww^{n+1} && \text{(definição de potência)} \\ &= w(w^n w) && \text{(hipótese de indução)} \\ &= (ww^n)w && \text{(associatividade)} \\ &= w^{n+1}w && \text{(definição de potência).} \end{aligned}$$

1.4.13. Partindo da definição indutiva de palavra reversa do Exemplo 1.3.4, mostre que $\forall a \in \Sigma, \forall v \in \Sigma^\star, (av)^{-1} = v^{-1}a$.

Resolução. Sejam $a \in \Sigma$ e $v \in \Sigma^\star$. Vamos usar indução estrutural sobre v para demonstrar a propriedade.

Se $v = \varepsilon$ então $(av)^{-1} = (a\varepsilon)^{-1} = a^{-1} = a = \varepsilon a = \varepsilon^{-1}a = v^{-1}a$.

Senão existem $b \in \Sigma$ e $u \in \Sigma^\star$ tais que $v = ub$. Vamos admitir que a propriedade se verifica para a e u, isto é, $(au)^{-1} = u^{-1}a$. Assim, $(av)^{-1} = (a(ub))^{-1} = ((au)b)^{-1} = b(au)^{-1} = b(u^{-1}a) = (bu^{-1})a = (ub)^{-1}a = v^{-1}a$, como queríamos demonstrar.

1.4.14. Seja Σ um qualquer alfabeto. Demonstre as propriedades seguintes.

(a) $\forall w \in \Sigma^\star, \forall n \in \mathbb{N}_0,\ |w^n| = n|w|$.

(b) $\forall x, y \in \Sigma^\star, (xy)^{-1} = y^{-1}x^{-1}$.

(c) $\forall w \in \Sigma^\star, (w^{-1})^{-1} = w$.

(d) $\forall w \in \Sigma^\star, \forall n \in \mathbb{N}_0,\ (w^n)^{-1} = (w^{-1})^n$.

(e) $\forall n \in \mathbb{N}, \forall x, y \in \Sigma^\star,\ x^n = y^n \implies x = y$.

(f) $\forall n \in \mathbb{N}_0, \forall w \in \Sigma^\star,\ w = w^{-1} \implies w^n = (w^n)^{-1}$.

(g) $\forall n \in \mathbb{N}, \forall w \in \Sigma^\star,\ w^n = (w^n)^{-1} \implies w = w^{-1}$.

Sugestão.

(a) A propriedade demonstra-se por indução, usando a definição de potência de uma palavra (Exemplo 1.3.1) e a propriedade aditiva do tamanho de palavras relativamente à concatenação (Exemplo 1.3.7).

(b) A propriedade demonstra-se por indução estrutural sobre x, usando a propriedade referida no Exercício 1.4.13 ou por indução estrutural sobre y, usando a definição indutiva de palavra reversa apresentada no Exemplo 1.3.4.

(c) A propriedade demonstra-se por indução estrutural, usando a propriedade referida no Exercício 1.4.13 e a definição indutiva de palavra reversa apresentada no Exemplo 1.3.4.

(d) A propriedade demonstra-se por indução sobre n, usando a propriedade referida na alínea (b) e a definição indutiva de potência de uma palavra.

(e) Basta utilizar a definição de igualdade de palavras em termos da igualdade de prefixos.

(f) É consequência da propriedade referida na alínea (d).

(g) É consequência das propriedades referidas nas alíneas (d) e (e).

1.4.15. Seja L uma linguagem com pelo menos duas palavras. Mostre que $L^2 \neq \{w^2 : w \in L\}$.

Resolução.

Caso 1. Suponhamos que existem duas palavras em L com o mesmo tamanho, digamos, $w_1 \neq w_2$ e $|w_1| = |w_2|$. Vamos mostrar que $w_1 w_2 \neq w^2$ para qualquer palavra $w \in L$.

De facto, admitindo que existia uma palavra $w \in L$ tal que $w_1 w_2 = w^2$ então, pela definição de igualdade de palavras, viria que $w_1 = w$ e $w_2 = w$. Assim, teríamos $w_1 = w_2$, o que é uma contradição.

Caso 2. Suponhamos que não existem em L duas palavras do mesmo tamanho.

Nesse caso, podemos ordenar as palavras da linguagem L por ordem de tamanho crescente, de tal modo que $w_1 \neq w_2$, $|w_1| < |w_2|$ e nenhuma palavra de L tem tamanho intermédio entre $|w_1|$ e $|w_2|$.

Podemos garantir que na linguagem $\{w^2 : w \in L\}$ não existem palavras de tamanho intermédio entre $|w_1^2|$ e $|w_2^2|$.

Por outro lado, podemos verificar que a palavra $w_1 w_2$ tem tamanho intermédio, isto é, $|w_1^2| < |w_1 w_2| < |w_2^2|$, pelo que $w_1 w_2 \notin \{w^2 : w \in L\}$.

1.4.16. Seja L uma linguagem com pelo menos duas palavras. Mostre que para $n \geq 2$, $L^n \neq \{w^n : w \in L\}$.

Resolução. É uma simples generalização da demonstração realizada no exercício anterior.

Caso 1. Suponhamos que existem duas palavras em L com o mesmo tamanho, digamos, $w_1 \neq w_2$ e $|w_1| = |w_2|$. Vamos mostrar que $w_1 w_2^{n-1} \neq w^n$ para qualquer palavra $w \in L$.

De facto, admitindo que existia uma palavra $w \in L$ tal que $w_1 w_2^{n-1} = w^n$ então, pela definição de igualdade de palavras, viria que $w_1 = w$ e $w_2 = w$. Assim, teríamos $w_1 = w_2$, o que é uma contradição.

Caso 2. Suponhamos que não existem em L duas palavras do mesmo tamanho.

Nesse caso, podemos ordenar as palavras da linguagem L por ordem de tamanho crescente, de tal modo que $w_1 \neq w_2$, $|w_1| < |w_2|$ e nenhuma palavra de L tem tamanho intermédio entre $|w_1|$ e $|w_2|$.

Podemos então garantir que na linguagem $\{w^n : w \in L\}$ não existem palavras de tamanho intermédio entre $|w_1^n| = n|w_1|$ e $|w_2^n| = n|w_2|$.

Por outro lado, podemos verificar que a palavra $w_1 w_2^{n-1}$ tem tamanho intermédio $|w_1| + (n-1)|w_2|$, isto é, $|w_1^n| < |w_1 w_2^{n-1}| < |w_2^n|$, pelo que $w_1 w_2^{n-1} \notin \{w^n : w \in L\}$.

1.4.17. Considere a definição de palavras conjugadas apresentada na Definição 1.1.17. Mostre que a relação de conjugação entre palavras é uma relação de equivalência.

Resolução. Seja $\sim$ a relação em $\Sigma^\star$ definida por $v \sim w$ se e só se v é conjugada de w.

Reflexividade. Queremos provar que $v \sim v$, qualquer que seja a palavra v.

Sejam $x = \varepsilon$ e $y = v$. Então $v = \varepsilon v = xy$ e $v = v\varepsilon = yx$. Logo, v é conjugada de si mesmo.

Simetria. Queremos provar que sempre que $v \sim w$ então também $w \sim v$, quaisquer que sejam as palavras v e w.

Se v é conjugada de w então existem x e y tais que $v = xy$ e $w = yx$. Assim, também $w = yx$ e $v = xy$ pelo que w é conjugada de v.

Transitividade. Queremos provar que sempre que $u \sim v$ e $v \sim w$ então também $u \sim w$, para quaisquer palavras u, v e w.

Pela definição de palavras conjugadas, se u é conjugada de v então existem x e y tais que $u = xy$ e $v = yx$. Também, se v é conjugada de w então existem x' e y' tais que $v = x'y'$ e $w = y'x'$.

Vamos assumir que $|x'| \leq |y|$. O caso contrário demonstra-se de forma análoga, pelo que se omite.

Uma vez que $yx = x'y'$, existe z tal que $y = x'z$ e $y' = zx$, donde $u = xx'z$ e $w = zxx'$. Assim, existem $x'' = xx'$ e $y'' = z$ tais que $u = x''y''$ e $w = y''x''$, pelo que u é conjugada de w.

1.4.18. Uma palavra w é livre de quadrados se não contém sub-palavras da forma uu, onde $u \neq \varepsilon$.

(a) Quantas palavras binárias livres de quadrados existem?

Resolução. O número de palavras sobre o alfabeto $\Sigma = \{0, 1\}$ que são livres de quadrados é finito. De facto, as palavras que não contêm o padrão uu são: ε, 0, 1, 01, 10, 010 e 101. Para além disso, estas palavras não podem

ser estendidas de modo a manter a propriedade de serem livres de quadrados. Por exemplo, ao estender a palavra 010 obtemos as palavras 0100 (contém 0^2) e 0101 (contém $(01)^2$).

(b) *** Mostre que existe um número infinito de palavras livres de quadrados definidas sobre o alfabeto $\Sigma = \{a, b, c\}$.

Resolução. Este problema foi originalmente estudado, entre 1906-1912, pelo matemático Axel Thue (Noruega, 1863 - 1922) e mais tarde referido e ampliado, em 1921, por Harold Morse (Estados Unidos da América, 1892 - 1977). Muitos outros artigos foram posteriormente publicados sobre o assunto e as suas generalizações. Para mais detalhes, consultar, por exemplo, [4].

Uma forma de gerar um número infinito de palavras livres de quadrados é baseada no homomorfismo $h\colon \{a, b, c\}^\star \to \{a, b, c\}^\star$ definido por $h(a) = abcab$, $h(b) = acabcb$ e $h(c) = acbcacb$. Aplicando sucessivamente o homomorfismo h a palavras livres de quadrados, obtemos novas palavras livres de quadrados de tamanho cada vez maior.

1.4.19. Mostre que dois homomorfismos de linguagens $h_1, h_2\colon \Sigma^\star \to \Gamma^\star$ são iguais se e só se $\forall a \in \Sigma$, $h_1(a) = h_2(a)$.

Resolução. Se os homomorfismos h_1 e h_2 são iguais então para qualquer palavra $w \in \Sigma^\star$, $h_1(w) = h_2(w)$. Em particular, para qualquer $a \in \Sigma \subset \Sigma^\star$, $h_1(a) = h_2(a)$.

Considerando que $h_1(a) = h_2(a)$, para qualquer $a \in \Sigma$, vamos mostrar por indução que $h_1(w) = h_2(w)$, para qualquer $w \in \Sigma^\star$.

Caso base. Para $w = \varepsilon$ então, por definição de homomorfismo, $h_1(w) = h_1(\varepsilon) = \varepsilon = h_2(\varepsilon) = h_2(w)$.

Indução estrutural. Seja $w = ax$, para algum símbolo $a \in \Sigma$ e para alguma palavra $x \in \Sigma^\star$ e admitamos por hipótese que $h_1(x) = h_2(x)$. Temos que:

$$\begin{aligned} h_1(w) = h_1(ax) &= h_1(a)h_1(x) && \text{(definição de homomorfismo)} \\ &= h_2(a)h_2(x) && \text{(hipótese de indução)} \\ &= h_2(ax) = h_2(w) && \text{(definição de homomorfismo)} \end{aligned}$$

1.4.20. Demonstre o Teorema 1.2.14.

Resolução. Apresentamos a demonstração apenas para a concatenação de linguagens. Para a união de linguagens procede-se de forma semelhante.

Sejam L_1 e L_2 duas quaisquer linguagens sobre o alfabeto Σ e seja $h\colon \Sigma^\star \to \Gamma^\star$ um homomorfismo de palavras. Vamos demonstrar a igualdade pretendida, verificando em primeiro lugar que $h(L_1L_2) \subseteq h(L_1)h(L_2)$ e, em seguida, que $h(L_1)h(L_2) \subseteq h(L_1L_2)$.

$\subseteq$. Seja $w \in L_1L_2$. Existem $x \in L_1$ e $y \in L_2$ tais que $w = xy$. Uma vez que h é um homomorfismo, $h(w) = h(xy) = h(x)h(y)$. Temos que $h(x) \in h(L_1)$ e $h(y) \in h(L_2)$, uma vez que $x \in L_1$ e $y \in L_2$. Logo, $h(w) = h(x)h(y) \in h(L_1)h(L_2)$. Concluímos que $h(L_1L_2) \subseteq h(L_1)h(L_2)$.

$\supseteq$. Seja $w \in h(L_1)h(L_2)$. Existem $x \in h(L_1)$ e $y \in h(L_2)$ tais que $w = xy$. Dado que $x \in h(L_1)$ e $y \in h(L_2)$, existe $u \in L_1$ tal que $x = h(u)$ e existe $v \in L_2$ tal que $y = h(v)$. Uma vez que h é um homomorfismo e atendendo a que $uv \in L_1L_2$, podemos afirmar que $w = xy = h(u)h(v) = h(uv) \in h(L_1L_2)$. Concluímos que $h(L_1)h(L_2) \subseteq h(L_1L_2)$.

1.4.21. Demonstre o Teorema 1.2.15.

Resolução. Vamos aplicar o Teorema 1.2.14 para provar que $h(L^n) = h(L)^n$, para qualquer $n \in \mathbb{N}_0$.

Caso base. Para $n = 0$, temos que $h(L^0) = h(\{\varepsilon\}) = \{h(\varepsilon)\} = \{\varepsilon\} = h(L)^0$.

Indução. Vamos considerar como hipótese de indução que $h(L^n) = h(L)^n$ e vamos provar que $h(L^{n+1}) = h(L)^{n+1}$.

De facto, temos que $h(L^{n+1}) = h(LL^n) = h(L)h(L^n) = h(L)h(L)^n = h(L)^{n+1}$, como queríamos demonstrar.

Assim, concluímos que

$$h(L^\star) = h(\cup_{n=0}^{\infty} L^n) = \cup_{n=0}^{\infty} h(L^n) = \cup_{n=0}^{\infty} h(L)^n = h(L)^\star.$$

1.4.22. Sejam $h\colon \Sigma^\star \to \Sigma^\star$ um homomorfismo de palavras e L uma linguagem sobre o alfabeto Σ. Mostre que $h(h^{-1}(L)) \subseteq L$ e que $L \subseteq h^{-1}(h(L))$.

Resolução.

$\subseteq$. Seja $w \in h(h^{-1}(L))$. Então existe $u \in h^{-1}(L)$ tal que $w = h(u)$. Mas $u \in h^{-1}(L)$, significa que $h(u) \in L$. Logo, $w \in L$. Concluímos que $h(h^{-1}(L)) \subseteq L$.

$\supseteq$. Seja agora $w \in L$. Então $h(w) \in h(L)$ e, pela definição de imagem inversa, $w \in h^{-1}(h(L)$. Concluímos que $L \subseteq h^{-1}(h(L))$.

1.4.23. Averigue se a classe das linguagens livres de prefixos é fechada para cada uma seguintes operações:

(a) união;

Resolução. Não é fechada para a união. Por exemplo, $L_1 = \{a\}$ e $L_2 = \{aa\}$ são livres de prefixos mas $L_1 \cup L_2 = \{a, aa\}$ não é livre de prefixos.

(b) intersecção;

Resolução. É fechada para a intersecção. Se L_1 e L_2 são livres de prefixos então $L_1 \cap L_2 \subseteq L_1$ também é livre de prefixos por maioria de razão.

(c) complementar;

Resolução. Não é fechada para o complementar. Por exemplo, considerando o alfabeto $\Sigma = \{a\}$ e a linguagem $L = \{\varepsilon\}$, vem que $\overline{L} = \Sigma^+ = \{a, aa, aaa, \ldots\}$ não é livre de prefixos.

(d) concatenação;

Resolução. É fechada para a concatenação. Sejam L_1 e L_2 duas linguagens livres de prefixos.

Se $L_1 = \emptyset$ ou $L_2 = \emptyset$ então $L_1L_2 = \emptyset$ é livre de prefixos. Assim, vamos considerar que tanto L_1 como L_2 são linguagens não vazias.

Se $L_1 = \{\varepsilon\}$ então $L_1L_2 = L_2$ é livre de prefixos. De igual modo, se $L_2 = \{\varepsilon\}$ então $L_1L_2 = L_1$ é livre de prefixos. Vamos então considerar que $L_1 \neq \{\varepsilon\}$ e $L_2 \neq \{\varepsilon\}$.

Se $L_1 \neq \{\varepsilon\}$ então $\varepsilon \notin L_1$, caso contrário L_1 não seria livre de prefixos. Pela mesma razão, $\varepsilon \notin L_2$.

Vamos admitir que existe uma palavra $xy \in L_1L_2$, com $x \in L_1$ e $y \in L_2$, tal que xy tem um prefixo próprio $uv \in L_1L_2$, com $u \in L_1$ e $v \in L_2$. Nesse caso $u = x$ e v é um prefixo próprio de y, pelo que L_2 não é livre de prefixos, o que é uma contradição. Concluímos que não existe qualquer palavra em L_1L_2 que tenha um prefixo próprio em L_1L_2, pelo que L_1L_2 é livre de prefixos.

(e) diferença;

Resolução. É fechada para a diferença de linguagens. Se L_1 e L_2 são livres de prefixos então $L_1 \setminus L_2 \subseteq L_1$ também é livre de prefixos por maioria de razão.

(f) potência;

Resolução. É fechada para a potência de linguagens. Se L é uma linguagem livre de prefixos então, como caso particular da concatenação de duas linguagens, decorre que $L^2 = LL$ também é livre de prefixos. Usando indução natural mostra-se que L^n, para $n \in \mathbb{N}$ também é livre de prefixos.

(g) fecho de Kleene.

Resolução. Não é fechada para o fecho de Kleene. Se $L \neq \emptyset$ então $\varepsilon \in L^\star$ e $L^\star \neq \{\varepsilon\}$. Uma vez que $\{\varepsilon\}$ é a única linguagem livre de prefixos que contém ε, $L^\star$ não pode ser livre de prefixos.

2 Autómatos finitos

2.6 Exercícios

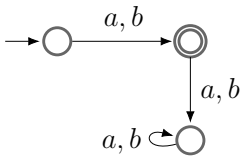
2.6.1. Modifique o autómato apresentado no Exemplo 2.1.1, de modo que as formas admitidas para comentários sejam apenas $(* \ldots *)$ e $\{\ldots\}$, pelo que não são admitidas as formas $(* \ldots \}$ e $\{\ldots *)$.

Sugestão. Basta duplicar o estado *comentário* separando o reconhecimento da forma $(* \ldots *)$ do reconhecimento da forma $\{\ldots\}$.

2.6.2. Construa autómatos finitos deterministas completos que reconheçam as seguintes linguagens definidas sobre o alfabeto $\Sigma = \{a, b\}$:

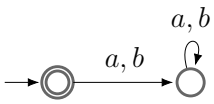
(a) $\{w \in \Sigma^\star : \ w \text{ tem uma letra}\}$

Solução.



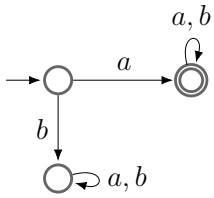
(b) $\{w \in \Sigma^\star : \ w \text{ não tem letras}\}$

Solução.



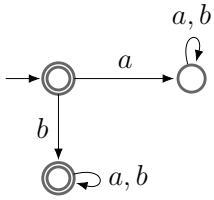
(c) $\{w \in \Sigma^\star : \ w \text{ começa com } a\}$

Solução.



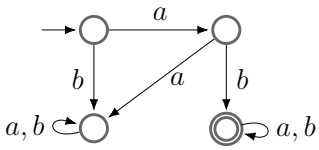
(d) $\{w \in \Sigma^\star : \ w \text{ não começa com } a\}$

Solução.



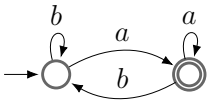
(e) $\{w \in \Sigma^\star : \ w \text{ começa com } ab\}$

Solução.



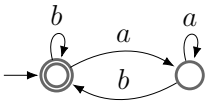
(f) $\{w \in \Sigma^\star : \ w \text{ acaba em } a\}$

Solução.



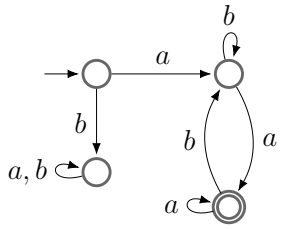
(g) $\{w \in \Sigma^\star : \ w \text{ não acaba em } a\}$

Solução.



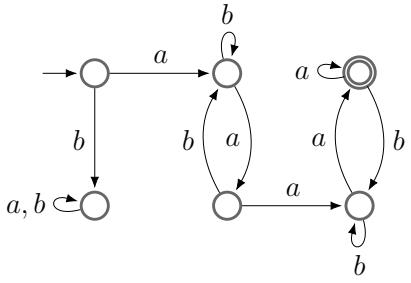
(h) $\{awa : \ w \in \Sigma^\star\}$

Solução.



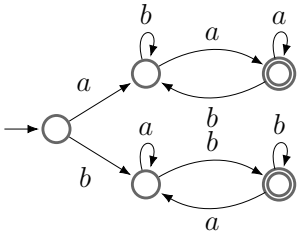
(i) $\{axaaya : x, y \in \Sigma^\star\}$

Solução.



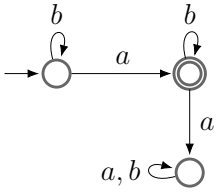
(j) $\{xwx \in \Sigma^\star : w \in \Sigma^\star, x \in \Sigma\}$

Solução.



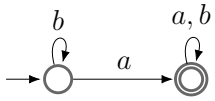
(k) $\{w \in \Sigma^\star : w$ tem exactamente um $a\}$

Solução.



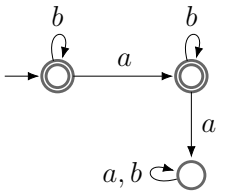
(l) $\{w \in \Sigma^\star : w$ tem pelo menos um $a\}$

Solução.



(m) $\{w \in \Sigma^\star : \ w \text{ tem no máximo um } a\}$

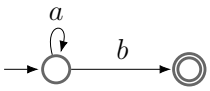
Solução.



2.6.3. Determine diagramas de transições de autómatos finitos deterministas que reconheçam as seguintes linguagens definidas sobre o alfabeto $\Sigma = \{a, b\}$:

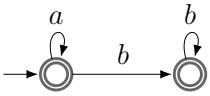
(a) $\{a^n b : \ n \geq 0\}$

Solução.



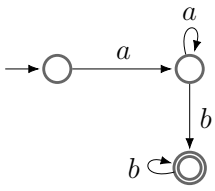
(b) $\{a^n b^m : \ n, m \geq 0\}$

Solução.



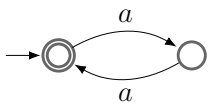
(c) $\{a^n b^m : \ n, m > 0\}$

Solução.



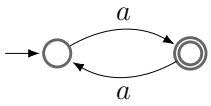
(d) $\{a^n : \ n \text{ é par}\}$

Solução.



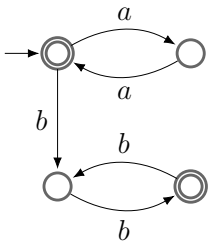
(e) $\{a^n : \ n \text{ é ímpar}\}$

Solução.



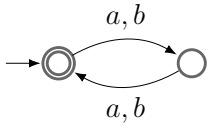
(f) $\{a^n b^m : \ m \text{ e } n \text{ são números pares}\}$

Solução.



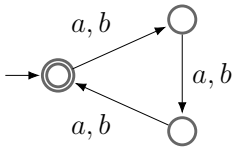
(g) $\{w \in \{a,b\}^\star : \ |w| \text{ é par}\}$

Solução.



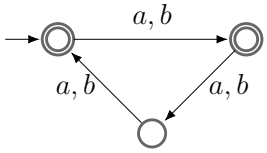
(h) $\{w \in \{a,b\}^\star : \ |w| \bmod 3 = 0\}$

Solução.



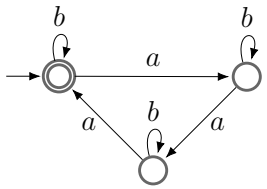
(i) $\{w \in \{a,b\}^\star : \ |w| \bmod 3 < 2\}$

Solução.



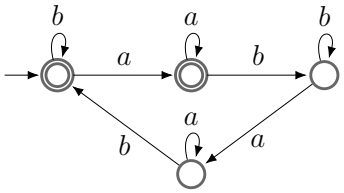
(j) $\{w \in \{a,b\}^\star : \ \#_a(w) \bmod 3 = 0\}$

Solução.



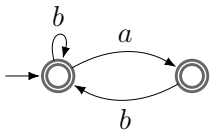
(k) $\{w \in \{a,b\}^\star : \ \#_{ab}(w) \bmod 2 = 0\}$

Solução.



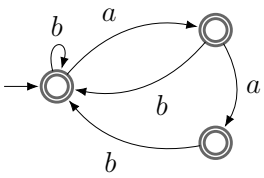
(l) $\{w \in \{a,b\}^\star : \ w$ não tem a's consecutivas$\}$

Solução.



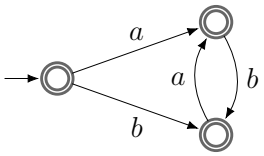
(m) $\{w \in \{a,b\}^\star : \ w$ não tem três a's consecutivos$\}$

Solução.



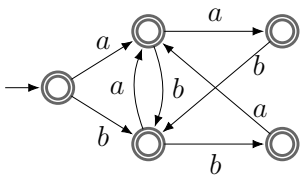
(n) $\{w \in \{a,b\}^\star : \ w$ não tem duas letras iguais consecutivas$\}$

Solução.



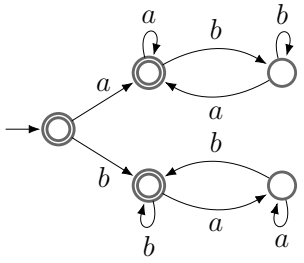
(o) $\{w \in \{a,b\}^\star : \ w$ não tem três letras iguais consecutivas$\}$

Solução.

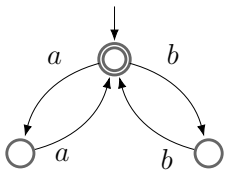


(p) $\{w \in \{a,b\}^\star : \ \#_{ab}(w) = \#_{ba}(w)\}$

Solução.



2.6.4. Caracterize a linguagem reconhecida pelo seguinte AFD:

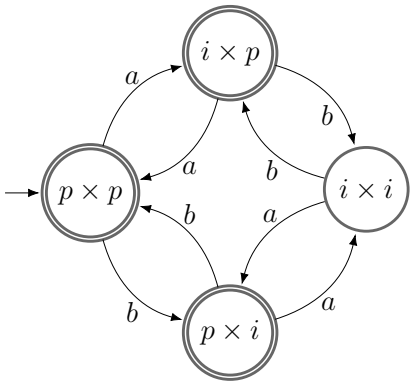


Solução. O autómato reconhece a linguagem $L = \{aa, bb\}^\star$.

2.6.5. Determine um AFD reconhecedor da linguagem

$$\{w \in \{a,b\}^\star : \ \#_a(w)\#_b(w) \text{ é par}\}.$$

Solução.



2.6.6. Mostre que qualquer AFD incompleto pode ser transformado num AFD completo que reconhece a mesma linguagem.

Resolução. Seja $A = \big(Q_A, \Sigma, \delta_A, s_A, F_A\big)$ um AFD incompleto. Para o completar, basta começar por acrescentar um novo estado ratoeira, r, e, em seguida, extender a função de transição parcial de tal modo que $\delta_B(q, a) = r$, sempre que $\delta_A(q, a)$ não esteja definido.

Este novo autómato, $B = \big(Q_A \cup \{r\}, \Sigma, \delta_B, s_A, F_A\big)$, reconhece todas as palavras reconhecidas pelo AFD original e apenas estas. De facto, $\delta_B(r, a) = r$, para qualquer $a \in \Sigma$, e $r \notin F_A$.

2.6.7. Aplique a fórmula (2.4.1) para desenvolver a função de transição estendida do autómato não determinista do Exemplo 2.4.7 com a palavra 0101.

Confirme que obtém o mesmo resultado que o que se obtém com a fórmula (2.4.2).

Resolução. Temos que

$$\begin{aligned}
\delta^\star(q_0, 0101) &= \text{fecho}\varepsilon(\delta(\delta^\star(q_0, 010), 1)) \\
\delta^\star(q_0, 010) &= \text{fecho}\varepsilon(\delta(\delta^\star(q_0, 01), 0)) \\
\delta^\star(q_0, 01) &= \text{fecho}\varepsilon(\delta(\delta^\star(q_0, 0), 1)) \\
\delta^\star(q_0, 0) &= \text{fecho}\varepsilon(\delta(\delta^\star(q_0, \varepsilon), 0)) \\
\delta^\star(q_0, \varepsilon) &= \text{fecho}\varepsilon(q_0) = \{q_0\}.
\end{aligned}$$

Assim,

$$\begin{aligned}
\delta^\star(q_0, 0) &= \text{fecho}\varepsilon(\delta(\{q_0\}, 0)) \\
&= \text{fecho}\varepsilon(\{q_1\}) = \{q_1, q_2\} \\
\delta^\star(q_0, 01) &= \text{fecho}\varepsilon(\delta(\{q_1, q_2\}, 1)) \\
&= \text{fecho}\varepsilon(\{q_1\}) = \{q_1, q_2\} \\
\delta^\star(q_0, 010) &= \text{fecho}\varepsilon(\delta(\{q_1, q_2\}, 0))
\end{aligned}$$

$$\begin{aligned} &= \text{fecho}\varepsilon(\{q_3\}) = \{q_1, q_2, q_3, q_4\} \\ \delta^\star(q_0, 0101) &= \text{fecho}\varepsilon(\delta(\{q_1, q_2, q_3, q_4\}, 1)) \\ &= \text{fecho}\varepsilon(\{q_1, q_3\}) = \{q_1, q_2, q_3, q_4\}. \end{aligned}$$

2.6.8. Considere o AFNDε do Exemplo 2.4.7.

(a) Mostre que todas as palavras começadas por 1 são rejeitadas.

Resolução. Seja $1x$, com $x \in \{0,1\}^\star$, uma qualquer palavra começada por 1. Aplicando a fórmula (2.4.2), obtemos

$$\delta^\star(q_0, 1x) = \delta^\star(\delta(\text{fecho}\varepsilon(q_0), 1), x) = \delta^\star(\delta(\{q_0\}, 1), x) = \delta^\star(\emptyset, x) = \emptyset.$$

Logo, a palavra $1x$ é rejeitada.

(b) O autómato aceita a palavra vazia?

Resolução. O autómato não aceita a palavra vazia, uma vez que $\delta^\star(q_0, \varepsilon) = \text{fecho}\varepsilon(q_0) = \{q_0\}$ e $\{q_0\} \cap F = \emptyset$.

(c) Qual é a linguagem do autómato?

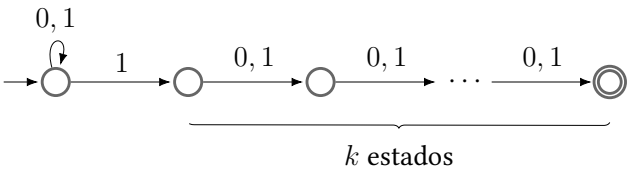
Resolução. A linguagem reconhecida pelo autómato é a das palavras binárias começadas pelo dígito 0 e com pelo menos dois dígitos 0 (incluindo o inicial).

2.6.9. Para $k \in \mathbb{N}$ considere a linguagem

$$L_k = \{w \in \{0,1\}^\star : \text{ o } k \text{ -ésimo dígito a contar da direita é } 1\}.$$

(a) Construa um AFND reconhecedor de L_k com $k+1$ estados.

Solução.



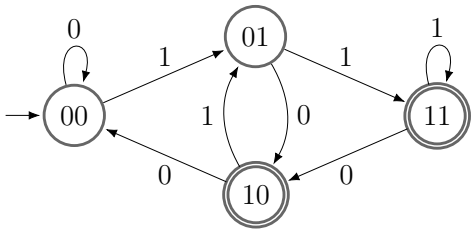
(b) Construa um AFD reconhecedor de L_k com 2^k estados. (Sugestão: associe a cada estado uma palavra de k dígitos.)

Resolução. Cada um dos 2^k estados representa uma palavra binária de tamanho k, associada à sequência dos últimos k dígitos lidos.

A partir do estado bw, com $b \in \{0,1\}$ e $w \in \{0,1\}^{k-1}$, o autómato transita por 0 para o estado $w0$ e transita por 1 para o estado $w1$.

Os 2^{k-1} estados de aceitação do autómato são da forma $1w$ com $w \in \{0,1\}^{k-1}$.

Por exemplo, para $k = 2$ obtemos o seguinte autómato:



(c) Mostre que qualquer AFD reconhecedor de L_k tem pelo menos 2^k estados.

Resolução. Vamos admitir que existe um AFD com menos do que 2^k estados que reconhece a linguagem L_k.

Uma vez que há 2^k palavras de tamanho k e o autómato tem menos do que 2^k estados, pelo princípio da gaiola dos pombos, existem necessariamente duas palavras distintas x e y, de tamanho k, cujo reconhecimento termina no mesmo estado, isto é, existe um estado q tal que $\delta^\star(s, x) = \delta^\star(s, y) = q$, onde s denota o estado inicial do AFD.

Seja i a primeira posição a contar da esquerda na qual as palavras x e y diferem. Vamos assumir, sem perda de generalidade, que x tem um 0 na posição i e que y tem um 1 na posição i.

Notamos que $x0^{i-1} \notin L_k$ e que $y0^{i-1} \in L_k$:

$$x0^{i-1} = \overbrace{\ldots}^{i-1}0\overbrace{\ldots}^{k-1}$$
$$y0^{i-1} = \overbrace{\ldots}^{i-1}1\overbrace{\ldots}^{k-1}$$

Sendo F o conjunto dos estados de aceitação do AFD, concluímos que $x0^{i-1} \notin F$ e que $y0^{i-1} \in F$. Em particular $\delta^\star(s, x0^{i-1}) \neq \delta^\star(s, y0^{i-1})$.

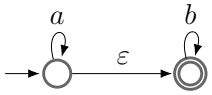
Por outro lado, uma vez que o autómato é determinista e $\delta^\star(s, x) = \delta^\star(s, y)$, também $\delta^\star(s, x0^{i-1}) = \delta^\star(s, y0^{i-1})$.

Chegamos assim a uma contradição, pelo que qualquer AFD que reconheça L_k tem de ter pelo menos 2^k estados.

2.6.10. Construa autómatos não deterministas, possivelmente com transições ε, que reconheçam as linguagens seguintes.

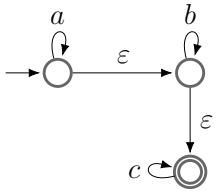
(a) $\{a^m b^n : \ m, n \geq 0\}$

Solução.



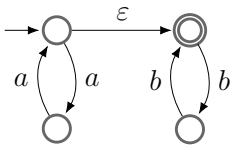
(b) $\{a^m b^n c^p : \ m, n, p \geq 0\}$

Solução.



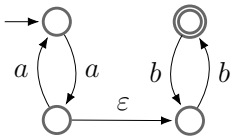
(c) $\{a^m b^n : \ m, n$ são pares$\}$

Solução.



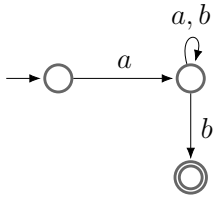
(d) $\{a^m b^n : \ m, n$ são ímpares$\}$

Solução.



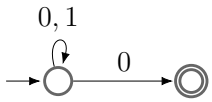
(e) $\{awb : \ w \in \{a, b\}^\star\}$

Solução.



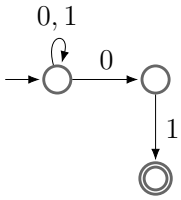
(f) $\{w \in \{0, 1\}^\star : \ w$ é a representação binária de um par$\}$

Solução.



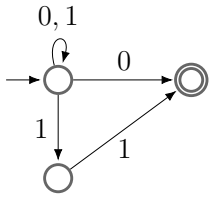
(g) $\{w \in \{0,1\}^\star : \ w \text{ acaba em } 01\}$

Solução.



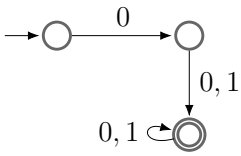
(h) $\{w \in \{0,1\}^\star : \ w \text{ acaba em } 0 \text{ ou em } 11\}$

Solução.



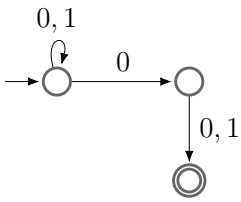
(i) $\{w \in \{0,1\}^\star : \ w \text{ começa com } 00 \text{ ou com } 01\}$

Solução.



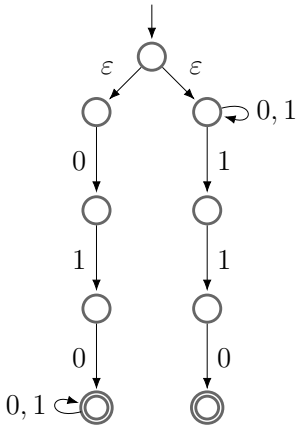
(j) $\{w \in \{0,1\}^\star : \ w \text{ acaba em } 00 \text{ ou em } 01\}$

Solução.



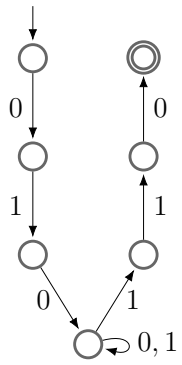
(k) $\{w \in \{0,1\}^\star : \ w \text{ começa com } 010 \text{ ou acaba em } 110\}$

Solução.



(l) $\{w \in \{0,1\}^\star : \ w \text{ começa com } 010 \text{ e acaba em } 110\}$

Solução.



2.6.11. Construa autómatos deterministas equivalentes aos seguintes autómatos não deterministas:

(a)

δ	0	1
$\to p$	$\{p, q\}$	$\{p\}$
q	$\{r, s\}$	$\{t\}$
r	$\{p, r\}$	$\{t\}$
$*s$	$\emptyset$	$\emptyset$
$*t$	$\emptyset$	$\emptyset$

Solução.

δ_D	0	1
$\to \{p\}$	$\{p,q\}$	$\{p\}$
$\{p,q\}$	$\{p,q,r,s\}$	$\{p,t\}$
$*\{p,t\}$	$\{p,q\}$	$\{p\}$
$*\{p,q,r,s\}$	$\{p,q,r,s\}$	$\{p,t\}$

(b)

δ	0	1
$\to p$	$\{p,q\}$	$\{p,r\}$
q	$\{r,s\}$	$\emptyset$
r	$\{p,r\}$	$\emptyset$
$*s$	$\{p\}$	$\{q\}$

Solução.

δ_D	0	1
$\to \{p\}$	$\{p,q\}$	$\{p,r\}$
$\{p,q\}$	$\{p,q,r,s\}$	$\{p,r\}$
$\{p,r\}$	$\{p,q,r\}$	$\{p,r\}$
$\{p,q,r\}$	$\{p,q,r,s\}$	$\{p,r\}$
$*\{p,q,r,s\}$	$\{p,q,r,s\}$	$\{p,q,r\}$

(c)

δ	a	b
$\to p$	$\{q\}$	$\{p\}$
q	$\{s\}$	$\{r,s\}$
r	$\{p,r\}$	$\{s\}$
$*s$	$\emptyset$	$\emptyset$

Solução.

δ_D	a	b
$\to \{p\}$	$\{q\}$	$\{p\}$
$\{q\}$	$\{s\}$	$\{r,s\}$
$*\{s\}$	$\emptyset$	$\emptyset$
$\{r,s\}$	$\{p,r\}$	$\{s\}$
$\{p,r\}$	$\{p,q,r\}$	$\{p,s\}$
$\{p,s\}$	$\{q\}$	$\{p\}$
$\{p,q,r\}$	$\{p,q,r,s\}$	$\{p,r,s\}$
$\{p,r,s\}$	$\{p,q,r\}$	$\{p,s\}$
$\{p,q,r,s\}$	$\{p,q,r,s\}$	$\{p,r,s\}$
$\emptyset$	$\emptyset$	$\emptyset$

(d)

δ	ε	a
$\to p$	$\emptyset$	$\{q\}$
$*q$	$\{r\}$	$\emptyset$
r	$\{p\}$	$\emptyset$

Solução.

δ_D	a
$\to \{p\}$	$\{p,q,r\}$
$*\{p,q,r\}$	$\{p,q,r\}$

(e)

δ	ε	0	1
$\to *p$	$\{r\}$	$\emptyset$	$\{q\}$
q	$\emptyset$	$\{p,r\}$	$\{r\}$
r	$\emptyset$	$\emptyset$	$\emptyset$

Solução.

δ_D	0	1
$\to *\{p,r\}$	$\emptyset$	$\{q\}$
$\{q\}$	$\{p,r\}$	$\{r\}$
$\{r\}$	$\emptyset$	$\emptyset$
$\emptyset$	$\emptyset$	$\emptyset$

(f)

δ	ε	0	1
$\to p$	$\{q\}$	$\{q\}$	$\emptyset$
$*q$	$\{r\}$	$\{p,r\}$	$\{q,r\}$
r	$\emptyset$	$\{r\}$	$\{q\}$

Solução.

δ_D	0	1
$\to *\{p,q,r\}$	$\{p,q,r\}$	$\{q,r\}$
$*\{q,r\}$	$\{p,q,r\}$	$\{q,r\}$

2.6.12. Determinize o seguinte AFNDε:

δ	ε	a	b	c
$\to p$	$\{q,r\}$	$\emptyset$	$\{q\}$	$\{r\}$
q	$\emptyset$	$\{p\}$	$\{r\}$	$\{p,q\}$
$*r$	$\emptyset$	$\emptyset$	$\emptyset$	$\{q\}$

Solução.

δ_D	a	b	c
$\to *\{p,q,r\}$	$\{p,q,r\}$	$\{q,r\}$	$\{p,q,r\}$
$*\{q,r\}$	$\{p,q,r\}$	$\{r\}$	$\{p,q,r\}$
$*\{r\}$	$\emptyset$	$\emptyset$	$\{q\}$
$\{q\}$	$\{p,q,r\}$	$\{r\}$	$\{p,q,r\}$

2.6.13.

(a) Mostre que para cada AFNDε existe um AFNDε equivalente com um único estado de aceitação.

Resolução. Seja $A = \big(Q_A, \Sigma, \delta_A, s_A, F_A\big)$ um AFNDε com mais do que um estado de aceitação. Construímos um novo autómato B a partir de A introduzindo um novo estado $f \notin Q_A$ e definindo transições ε dos estados em F_A para f. Os estados em F_A deixam de ser estados de aceitação.

Mais formalmente, $B = \big(Q_A \cup \{f\}, \Sigma, \delta_B, s_A, \{f\}\big)$, onde:

- para $q \in Q_A \setminus F_A$ e $a \in \Sigma \cup \{\varepsilon\}$, $\delta_B(q,a) = \delta_A(q,a)$;
- para $q \in F_A$ e $a \in \Sigma \cup \{\varepsilon\}$, $\delta_B(q,a) = \delta_A(q,a) \cup \{f\}$;
- para $a \in \Sigma \cup \{\varepsilon\}$, $\delta_B(f,a) = \emptyset$.

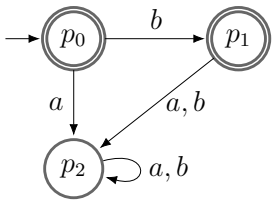
Os autómatos A e B são equivalentes, uma vez que para qualquer palavra $w \in \Sigma^\star$, $\delta_A^*(s_A, w) \in F_A$ se e só se $\delta_B^*(s_A, w) = f$.

(b) Será que qualquer qualquer AFD é equivalente a um AFD com apenas um estado de aceitação?

Resolução. Existem AFD que não são equivalentes a qualquer AFD com um único estado de aceitação. Em termos de linguagens, podemos dizer que existem linguagens que não são reconhecidas por qualquer AFD com um único estado de aceitação, embora sejam reconhecidas por AFD com mais do que um estado de aceitação.

Um exemplo simples, é dado pela linguagem finita $L = \{\varepsilon, b\}$.

Esta linguagem é reconhecida pelo seguinte AFD com dois estados de aceitação:



Por outro lado, se $A = \big(Q, \Sigma, \delta, s, F\big)$ é qualquer AFD que reconheçe L então, uma vez que $\varepsilon \in L$, o estado inicial s tem de ser um estado de aceitação. Para além disso, como $b \in L$, $\delta(s, b)$ tem de ser também um estado final e não pode ser igual a s, senão a palavra bb também seria reconhecida pelo autómato (na verdade qualquer palavra da forma b^k com $k \in \mathbb{N}$ seria reconhecida). Logo, o autómato A tem de possuir pelo menos dois estados de aceitação.

2.6.14. Sejam $A = \big(Q_A, \Sigma, \delta_A, s_A, F_A\big)$ e $B = \big(Q_B, \Sigma, \delta_B, s_B, F_B\big)$ dois AFD completos reconhecedores das linguagens L_A e L_B, respectivamente. Mostre que $L_A \cap L_B$ é reconhecida pelo AFD $A \cap B = \big(Q_A \times Q_B, \Sigma, \delta_{A\cap B}, s_{A\cap B}, F_{A\cap B}\big)$, onde:

- $s_{A\cap B} = (s_A, s_B)$;
- $F_{A\cap B} = F_A \times F_B$;
- para $a \in \Sigma$ e $(q_A, q_B) \in Q_A \times Q_B$,

$$\delta_{A\cap B}\big((q_A, q_B),\, a\big) = \big(\delta_A(q_A, a),\, \delta_B(q_B, a)\big).$$

Resolução.

(A). Seja $w \in L_A \cap L_B$. Então $w \in L_A$ e $w \in L_B$. Pela definição de palavra reconhecida por um AFD, ${\delta^\star}_A\,(s_A, w) \in F_A$ e ${\delta^\star}_B\,(s_B, w) \in F_B$. Assim, pela definição de produto cartesiano, $({\delta^\star}_A\,(s_A, w), {\delta^\star}_B\,(s_B, w)) \in F_A \times F_B$.

Prova-se por indução que, para qualquer estado $(q_A, q_B) \in Q_{A\cup B}$ e para qualquer palavra $w \in \Sigma^\star$,

$$\delta^\star{}_{A\cap B}\left((q_A, q_B), w\right) = \left(\delta^\star{}_A(q_A, w), \delta^\star{}_B(q_B, w)\right).$$

Assim,

$$\delta^\star{}_{A\cap B}\left((s_A, s_B), w\right) \in F_A \times F_B,$$

ou seja, $\delta^\star{}_{A\cap B}\left(s_{A\cap B}, w\right) \in F_{A\cap B}$ e portanto $w \in \mathcal{L}(A \cap B)$.

(B). Reciprocamente, se $w \in \mathcal{L}(A \cap B)$ então $\delta^\star{}_{A\cap B}\left(s_{A\cap B}, w\right) \in F_{A\cap B}$, ou seja,

$$\delta^\star{}_{A\cap B}\left((s_A, s_B), w\right) \in F_A \times F_B.$$

De $\delta^\star{}_{A\cap B}\left((s_A, s_B), w\right) = (\delta^\star{}_A(s_A, w), \delta^\star{}_B(s_B, w))$ concluímos que $\delta^\star{}_A\left(s_A, w\right) \in F_A$ e $\delta^\star{}_B\left(s_B, w\right) \in F_B$, ou seja, $w \in L_A \cap L_B$.

De (A) e (B) concluímos que $w \in L_A \cap L_B$ se e só se $w \in \mathcal{L}(A \cap B)$ e portanto $\mathcal{L}(A \cap B) = L_A \cap L_B$, como queríamos demonstrar.

2.6.15. Sejam $A = \big(Q_A, \Sigma, \delta_A, s_A, F_A\big)$ e $B = \big(Q_B, \Sigma, \delta_B, s_B, F_B\big)$ autómatos finitos deterministas completos reconhecedores das linguagens L_A e L_B, respectivamente. Mostre que $L_A \cup L_B$ é reconhecida pelo AFD $A \cup B = \big(Q_A \times Q_B,\ \Sigma,\ \delta_{A\cup B},\ s_{A\cup B},\ F_{A\cup B}\big)$, onde:

- $s_{A\cup B} = (s_A, s_B)$;
- $F_{A\cup B} = F_A \times Q_B \cup Q_A \times F_B$;
- para $a \in \Sigma$ e $(q_A, q_B) \in Q_A \times Q_B$,

$$\delta_{A\cup B}\big((q_A, q_B),\ a\big) = \big(\delta_A(q_A, a),\ \delta_B(q_B, a)\big).$$

Resolução.

(A). Seja $w \in L_A \cup L_B$. Então $w \in L_A$ ou $w \in L_B$. Pela definição de palavra reconhecida por um AFD, $\delta^\star{}_A(s_A, w) \in F_A$ ou $\delta^\star{}_B(s_B, w) \in F_B$. Assim, pela definição de produto cartesiano,

$$(\delta^\star{}_A(s_A, w), \delta^\star{}_B(s_B, w)) \in F_A \times Q_B \cup Q_A \times F_B.$$

Prova por indução que, para qualquer estado $(q_A, q_B) \in Q_{A\cup B}$ e palavra $w \in \Sigma^\star$,

$$\delta^\star{}_{A\cup B}\left((q_A, q_B), w\right) = \left(\delta^\star{}_A(q_A, w), \delta^\star{}_B(q_B, w)\right).$$

Assim,

$$\delta^\star{}_{A\cup B}\left((s_A, s_B), w\right) \in F_A \times Q_B \cup Q_A \times F_B,$$

ou seja, $\delta^\star{}_{A\cup B}\left(s_{A\cup B}, w\right) \in F_{A\cup B}$ e portanto $w \in \mathcal{L}(A \cup B)$.

(B). Reciprocamente, se $w \in \mathcal{L}(A \cup B)$ então $\delta^\star{}_{A\cup B}\left(s_{A\cup B}, w\right) \in F_{A\cup B}$, ou seja,

$$\delta^\star{}_{A\cup B}\left(\left(s_A, s_B\right), w\right) \in F_A \times Q_B \cup Q_A \times F_B.$$

Logo, $\delta^\star{}_{A\cup B}\left(\left(s_A, s_B\right), w\right) \in F_A \times Q_B$ ou $\delta^\star{}_{A\cup B}\left(\left(s_A, s_B\right), w\right) \in Q_A \times F_B$.

Uma vez que,

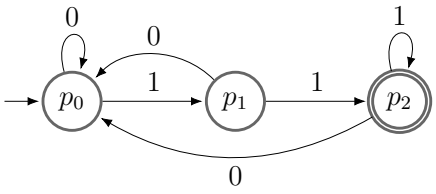
$$\delta^\star{}_{A\cup B}\left(\left(s_A, s_B\right), w\right) = \left(\delta^\star{}_A(s_A, w), \delta^\star{}_B(s_B, w)\right),$$

concluímos que $\delta^\star{}_A(s_A, w) \in F_A$ ou $\delta^\star{}_B(s_B, w) \in F_B$, ou seja, $w \in L_A \cup L_B$.

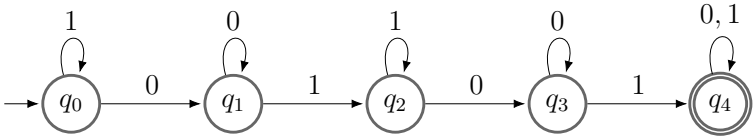
De (A) e (B) concluímos que $w \in L_A \cup L_B$ se e só se $w \in \mathcal{L}(A \cup B)$ e portanto $\mathcal{L}(A \cup B) = L_A \cup L_B$, como queríamos demonstrar.

2.6.16. Usando a construção do autómato produto, determine um AFD reconhecedor da linguagem das palavras binárias que têm pelo menos duas ocorrências de 01 e acabam em 11.

Resolução. Um AFD reconhecedor da linguagem das palavras binárias que acabam em 11 é



Um AFD reconhecedor da linguagem das palavras binárias que têm pelo menos duas ocorrências de 01 é



As tabelas de transições correspondentes são:

δ_A	0	1
$\rightarrow p_0$	p_0	p_1
p_1	p_0	p_2
$*p_2$	p_0	p_2

δ_B	0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_3	q_4
$*q_4$	q_4	q_4

A partir das tabelas de transições associadas a estes autómatos construímos a tabela

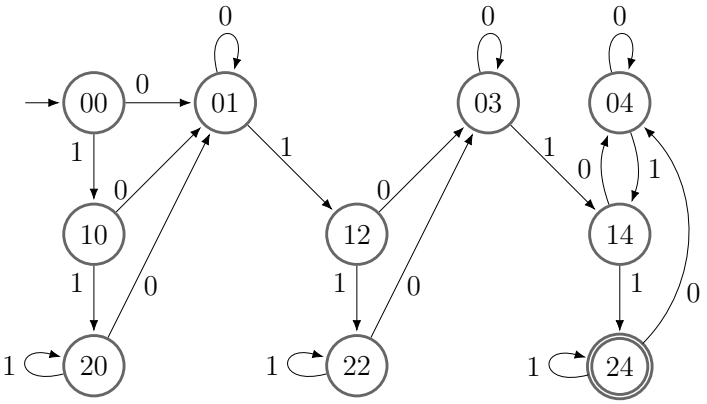
do autómato produto:

$\delta_{A\cap B}$	0	1
$\to p_0q_0$	p_0q_1	p_1q_0
p_0q_1	p_0q_1	p_1q_2
p_0q_2	p_0q_3	p_1q_2
p_0q_3	p_0q_3	p_1q_4
p_0q_4	p_0q_4	p_1q_4

$\delta_{A\cap B}$	0	1
p_1q_0	p_0q_1	p_2q_0
p_1q_1	p_0q_1	p_2q_2
p_1q_2	p_0q_3	p_2q_2
p_1q_3	p_0q_3	p_2q_4
p_1q_4	p_0q_4	p_2q_4

$\delta_{A\cap B}$	0	1
p_2q_0	p_0q_1	p_2q_0
p_2q_1	p_0q_1	p_2q_2
p_2q_2	p_0q_3	p_2q_2
p_2q_3	p_0q_3	p_2q_4
$*p_2q_4$	p_0q_4	p_2q_4

Dos 15 estados apenas 10 são acessíveis a partir do estado inicial e assim obtemos o autómato:



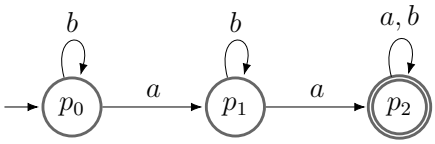
Veremos no capítulo seguinte que se pode simplificar ainda mais este autómato.

2.6.17. Usando a construção do autómato produto obtenha um AFD que reconheça a linguagem das palavras sobre o alfabeto $\Sigma = \{a, b\}$ que têm pelo menos dois a's ou no máximo dois b's.

Resolução. Um AFD reconhecedor da linguagem

$$\{w \in \{a,b\}^\star : \ w \text{ tem pelo menos dois } a\text{'s}\}$$

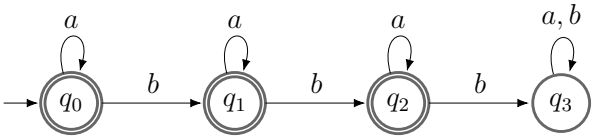
é



Um AFD reconhecedor da linguagem

$$\{w \in \{a,b\}^\star : \ w \text{ tem no máximo dois } b\text{'s}\}$$

é



As tabelas de transições correspondentes são:

δ_A	a	b
$\to p_0$	p_1	p_0
p_1	p_2	p_1
$*p_2$	p_2	p_2

δ_B	a	b
$\to *q_0$	q_0	q_1
$*q_1$	q_1	q_2
$*q_2$	q_2	q_3
q_3	q_3	q_3

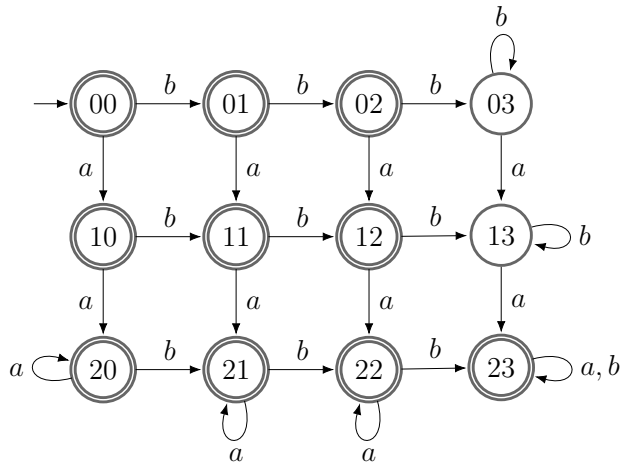
A partir das tabelas de transições associadas a estes autómatos construímos a tabela do autómato produto:

$\delta_{A\cup B}$	a	b
$\to *p_0q_0$	p_1q_0	p_0q_1
$*p_0q_1$	p_1q_1	p_0q_2
$*p_0q_2$	p_1q_2	p_0q_3
p_0q_3	p_1q_3	p_0q_3

$\delta_{A\cup B}$	a	b
$*p_1q_0$	p_2q_0	p_1q_1
$*p_1q_1$	p_2q_1	p_1q_2
$*p_1q_2$	p_2q_2	p_1q_3
p_1q_3	p_2q_3	p_1q_3

$\delta_{A\cup B}$	a	b
$*p_2q_0$	p_2q_0	p_2q_1
$*p_2q_1$	p_2q_1	p_2q_2
$*p_2q_2$	p_2q_2	p_2q_3
$*p_2q_3$	p_2q_3	p_2q_3

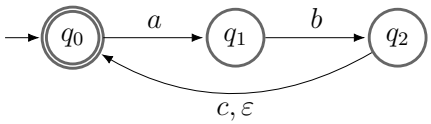
Veremos no capítulo seguinte que se pode simplificar este autómato, representado pelo seguinte diagrama:



2.6.18. Considere o alfabeto $\Sigma = \{a, b, c\}$.

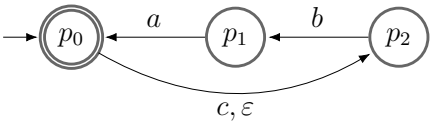
(a) Construa um AFNDε com apenas 3 estados e 4 transições que reconheça a linguagem $L = \{abc, ab\}^\star$.

Resolução. Um AFNDε é dado pelo seguinte diagrama de transições:

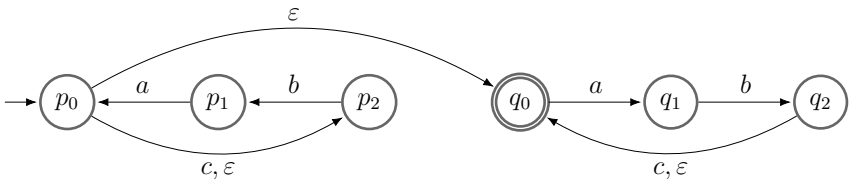


(b) Construa um AFNDε que reconheça a linguagem $L^{-1}L$.

Resolução. Um AFNDε que reconhece L^{-1} é:



Assim, um AFNDε que reconhece $L^{-1}L$ é:



(c) Aplicando o algoritmo de determinização, construa um AFD equivalente ao AFNDε obtido na alínea anterior.

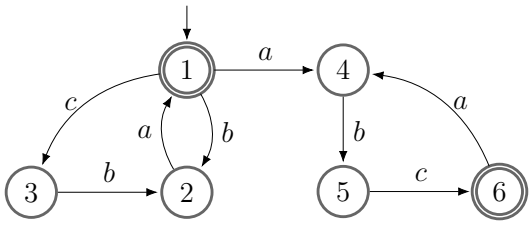
Resolução. A partir da tabela de transições do AFNDε

δ	ε	a	b	c
$\to p_0$	$\{p_2, q_0\}$	$\emptyset$	$\emptyset$	$\{p_2\}$
p_1	$\emptyset$	$\{p_0\}$	$\emptyset$	$\emptyset$
p_2	$\emptyset$	$\emptyset$	$\{p_1\}$	$\emptyset$
$*q_0$	$\emptyset$	$\{q_1\}$	$\emptyset$	$\emptyset$
q_1	$\emptyset$	$\emptyset$	$\{q_2\}$	$\emptyset$
q_2	$\{q_0\}$	$\emptyset$	$\emptyset$	$\{q_0\}$

construímos a tabela de transições de um AFD equivalente

δ	a	b	c
$\to *\{p_0, p_2, q_0\}$	$\{q_1\}$	$\{p_1\}$	$\{p_2\}$
$\{p_1\}$	$\{p_0, p_2, q_0\}$	$\emptyset$	$\emptyset$
$\{p_2\}$	$\emptyset$	$\{p_1\}$	$\emptyset$
$\{q_1\}$	$\emptyset$	$\{q_0, q_2\}$	$\emptyset$
$\{q_2\}$	$\emptyset$	$\emptyset$	$\{q_0\}$
$*\{q_0\}$	$\{q_1\}$	$\emptyset$	$\emptyset$

representado pelo seguinte diagrama:



2.6.19. Diga, justificando, qual é o valor lógico de cada uma das seguintes afirmações:

(a) Se $A = \big(Q,\, \Sigma,\, \delta,\, s,\, F\big)$ é um autómato finito não determinista que reconhece uma linguagem L então $A^c = \big(Q,\, \Sigma,\, \delta,\, s,\, Q \setminus F\big)$ é um autómato que reconhece a linguagem $\Sigma^\star \setminus L$.

(b) Se $A = \big(Q,\, \Sigma,\, \delta,\, s,\, F\big)$ é um autómato finito determinista que reconhece uma linguagem L então $A^c = \big(Q,\, \Sigma,\, \delta,\, s,\, Q \setminus F\big)$ é um autómato que reconhece a linguagem $\Sigma^\star \setminus L$.

Resolução. São ambas falsas. Em geral, o autómato $A = \big(Q,\, \Sigma,\, \delta,\, s,\, F\big)$ deve ser determinista e completo, conforme referido nos exemplos 2.1.14 e 2.2.11.

2.6.20. Sejam $A = \big(Q_A,\, \Sigma,\, \delta_A,\, s_A,\, F_A\big)$ e $B = (Q_B, \Sigma, \delta_B, s_B, F_B)$ AFD completos reconhecedores das linguagens L_A e L_B, respectivamente. Mostre que qualquer palavra da linguagem $L_A \setminus L_B$ é reconhecida pelo AFD

$$A \setminus B = \big(Q_A \times Q_B,\, \Sigma,\, \delta_{A\setminus B},\, s_{A\setminus B},\, F_{A\setminus B}\big)$$

, onde:

- $s_{A\setminus B} = (s_A, s_B)$;
- $F_{A\setminus B} = F_A \times (Q_B \setminus F_B)$;
- para $a \in \Sigma$ e $(q_A, q_B) \in Q_A \times Q_B$,

$$\delta_{A\setminus B}\big((q_A, q_B), a\big) = \big(\delta_A(q_A, a), \delta_B(q_B, a)\big).$$

Resolução. Seja $w \in L_A \setminus L_B$. Pela definição de conjunto diferença, $w \in L_A$ e $w \notin L_B$. Pela definição de palavra reconhecida por um AFD, $\delta^\star{}_A(s_A, w) \in F_A$ e $\delta^\star{}_B(s_B, w) \in Q_B \setminus F_B$. Assim, pela definição de produto cartesiano,

$$(\delta^\star{}_A(s_A, w), \delta^\star{}_B(s_B, w)) \in F_A \times (Q_B \setminus F_B)\,.$$

Prova-se por indução que, para qualquer estado $(q_A, q_B) \in Q_{A\times B}$ e palavra $w \in \Sigma^\star$,

$$\delta^\star{}_{A\setminus B}\left((q_A, q_B)\,, w\right) = (\delta^\star{}_A(q_A, w), \delta^\star{}_B(q_B, w))\,.$$

Assim, $\delta^\star{}_{A\setminus B}\left((s_A, s_B), w\right) \in F_A \times (Q_B \setminus F_B)$, ou seja, $\delta^\star{}_{A\setminus B}\left(s_{A\setminus B}, w\right) \in F_{A\setminus B}$ e portanto $w \in L_{A\setminus B}$.

2.6.21. Considere a seguinte proposta de extensão do método de construção do autómato produto para a união de linguagens de dois AFD referido no exercício Exercício 2.6.15, à classe dos AFND:

Sejam $A = \left(Q_A,\, \Sigma,\, \delta_A,\, s_A,\, F_A\right)$ e $B = (Q_B, \Sigma, \delta_B, s_B, F_B)$ AFND reconhecedores das linguagens L_A e L_B, respectivamente. Definimos o autómato produto

$$A \cup B = \left(Q_A \times Q_B,\, \Sigma,\, \delta_{A\cup B},\, s_{A\cup B},\, F_{A\cup B}\right)$$

, onde:

- $s_{A\cup B} = (s_A, s_B)$;
- $F_{A\cup B} = F_A \times Q_B \cup Q_A \times F_B$;
- para $a \in \Sigma$ e $(q_A, q_B) \in Q_A \times Q_B$,

$$\delta_{A\cup B}\big((q_A, q_B),\, a\big) = \delta_A(q_A, a) \times \delta_B(q_B, a).$$

Verifique que com esta definição nem sempre é verdade que $\mathcal{L}(A \cup B) = \mathcal{L}(A) \cup \mathcal{L}(B)$.

Resolução. Consideremos o AFND $A = (\{q_0\}, \{a, b\}, q_0, \delta_A, \{q_0\})$, onde $\delta_A(q_0, a) = \{q_0\}$ e $\delta_A(q_0, b) = \emptyset$, reconhecedor da linguagem $\mathcal{L}(A) = \{a^n : \ n \geq 0\}$, e o AFND $B = (\{p_0\}, \{a, b\}, p_0, \delta_B, \{p_0\})$, onde $\delta_B(p_0, a) = \emptyset$ e $\delta_B(p_0, b) = \{p_0\}$, reconhecedor da linguagem $\mathcal{L}(B) = \{b^n : \ n \geq 0\}$.

Segundo a definição proposta, o autómato produto é

$$A \cup B = (\{(q_0, p_0\}, \{a, b\}, (q_0, p_0), \delta_{A\cup B}, \{(q_0, p_0)\})$$

onde $\delta_{A\cup B}((q_1, q_2), a) = \emptyset$ e $\delta_{A\cup B}((q_1, q_2), b) = \emptyset$. Logo, $\mathcal{L}(A \cup B) = \{\varepsilon\} \neq \mathcal{L}(A) \cup \mathcal{L}(B)$.

2.6.22. Seja L a linguagem definida sobre um alfabeto Σ, reconhecida por um AFD A. Construa autómatos finitos reconhecedores das linguagens seguintes.

(a) Para $a \in \Sigma$ fixo, $a \,/\, L = \{x \in \Sigma^\star : \ ax \in L\}$;

Resolução. Seja $A = (Q, \Sigma, \delta, s, F)$ um AFD reconhecedor de L. Basta alterar o estado inicial do autómato A de s_A para $s_B = \delta_A(s, a)$.

Resta mostrar que o autómato $B = (Q, \Sigma, \delta, s_B, F)$, assim definido, reconhece a linguagem $a \,/\, L$, o que deixamos ao cuidado do leitor.

(b) Para $a \in \Sigma$ fixo, $L \,/\, a = \{x \in \Sigma^\star : \ xa \in L\}$;

Resolução. Seja $A = (Q, \Sigma, \delta_A, s_A, F_A)$ um AFD reconhecedor de L. Co-

meçamos por determinar os estados deste autómato que têm transições pelo símbolo a para um estado de aceitação: $P = \{q \in Q_A : \ \delta(q, a) \in F_A\}$.

Transformamos o AFD A num AFD B, começando por passar os estados de aceitação do autómato A a estados de não aceitação e, em seguida, passando os estados no conjunto P a estados de aceitação (neste processo um estado de aceitação pode voltar a ser estado de aceitação).

Resta mostrar que o autómato $B = (Q, \Sigma, \delta, s, P)$, assim definido, reconhece a linguagem $L \,/\, a$, o que deixamos ao cuidado do leitor.

Quando utilizada na área das expressões regulares, as quais serão estudadas no Capítulo 4, esta linguagem é chamada a derivada de L por a. Para mais detalhes sugerimos começar pelo trabalho pioneiro de Brzozowski (Januz A. Brzozowski, Polónia, 1935 - 2019) [13].

(c) metade$(L) = \{x \in \Sigma^\star : \ \text{ existe } y \in \Sigma^\star \text{ com } |x| = |y| \text{ e } xy \in L\}$.

Resolução. Seja $A = (Q, \Sigma, \delta_A, s_A, F_A)$ um AFD reconhecedor de L. O reconhecimento de uma palavra $xy \in L$ tal que $|x| = |y|$ consiste num caminho do estado inicial s_A para um determinado estado $q \in Q$, por leitura da primeira metade x, seguida de um caminho do estado q para um estado de aceitação, por leitura da segunda metade y.

Assim, vamos particionar a linguagem metade(L) em linguagens metade$(L)_q$, $q \in Q$, definidas por

$$\text{metade}(L)_q = \{x \in \Sigma^\star : \ \ x \in \text{metade}(L) \text{ e } \ \delta^\star{}_A(s_A, x) = q\}$$

e, em seguida, definir um autómato reconhecedor de cada uma delas.

Temos que metade$(L) = \bigcup_{q \in Q} \text{metade}(L)_q$, pelo que podemos usar a construção do produto de autómatos para obter um autómato reconhecedor da união destas linguagens.

Vamos então ver como construir um autómato para a linguagem metade$(L)_q$.

Para cada $q \in Q$, construímos o AFND B_q obtido a partir do autómato A pelo seguinte processo:

- o estado inicial do autómato B_q passa a ser o estado q;
- se existe uma transição de p para t no autómato A então o autómato B_q tem todas as transições da forma $\delta_{B_q}(p, b) = t$ para $b \in \Sigma$.

Notamos que este autómato aceita uma palavra x se e só se existe uma palavra y, do mesmo tamanho que o de x, tal que $\delta^\star{}_A(q, y) \in F_A$.

Para cada $q \in Q$, construímos ainda o AFD A_q obtido a partir do autómato A definindo q como o único estado de aceitação. Notamos que o autómato A_q reconhece uma palavra x se e só se $\delta^\star{}_A(s_A, x) = q$.

Temos assim que metade$(L)_q$ é a intersecção das linguagens reconhecidas pelos autómatos A_q e B_q, pelo que podemos usar a construção do produto de autómatos para obter um autómato reconhecedor da intersecção (após determinizar B_q).

3 Minimização de AFD

3.4 Exercícios

3.4.1. Elabore um algoritmo para calcular o conjunto de todos os estados inacessíveis de um autómato finito determinista.

Resolução. Seja $A = (Q, \Sigma, \delta, s, F)$ um AFD. Consideramos o seguinte algoritmo para calcular o conjunto dos estados acessíveis.

$X \leftarrow \{s\}$;

Repetir

$A \leftarrow \{\delta(q, a) : \ a \in \Sigma, q \in X\}$;

Se $X = X \cup A$

então STOP;

senão $X \leftarrow X \cup A$;

No final da execução do algoritmo, o conjunto $Q \setminus X$ é o conjunto dos estados inacessíveis do autómato.

3.4.2. Demonstre o Lema 3.2.3.

Resolução. Vamos demonstrar o resultado por indução estrutural sobre $\Sigma^\star$. Para $w = \varepsilon$ basta aplicar a definição de função de transição estendida, $\delta^\star{}_{\equiv}([q], \varepsilon) = [q]$ e $\delta^\star(q, \varepsilon) = q$. Para $w = a$, com $a \in \Sigma$, basta aplicar a definição de função de transição estendida, $\delta^\star{}_{\equiv}([q], a) = \delta_{\equiv}([q], a)$ e $\delta^\star(q, a) = \delta(q, a)$.

Supondo que a propriedade é válida para qualquer palavra x, mostramos que é válida ainda para $w = xa$, qualquer que seja $a \in \Sigma$.

$$\begin{aligned}
\delta^\star{}_{\equiv}([q], w) &= \delta^\star{}_{\equiv}([q], xa) \\
&= \delta_{\equiv}(\delta^\star{}_{\equiv}([q], x), a) && \text{(def. de função de transição estendida)} \\
&= \delta_{\equiv}([\delta^\star(q, x)], a) && \text{(hipótese de indução)} \\
&= [\delta(\delta^\star(q, x), a)] && \text{(def. de } \delta_{\equiv}\text{)} \\
&= [\delta^\star(q, xa)] = [\delta^\star(q, w)] && \text{(def. de } \delta^\star\text{)}
\end{aligned}$$

3.4.3. Mostre que a função h referida na demonstração do Lema 3.2.15 é bijectiva

Resolução.

Injectividade. Sejam $p, q \in Q_A$ tais que $p \neq q$. Como o autómato A é reduzido, concluímos que os estados p e q são distinguíveis. Logo, existe uma palavra x que os distingue, pelo que apenas uma das palavras $w_p x$ ou $w_q x$ pertence a $\mathcal{L}(A)$.

Como os autómatos A e B são equivalentes, podemos afirmar que apenas uma das palavras $w_p x$ ou $w_q x$ pertence a $\mathcal{L}(B)$ e, em particular, que $\delta^\star(s_B, w_p x) \neq \delta^\star(s_B, w_q x)$.

Como o autómato B é determinista, concluímos que $\delta^\star(s_B, w_p) \neq \delta^\star(s_B, w_q)$, pelo que $h(p) \neq h(q)$.

Sobrejectividade. Seja $p \in Q_B$. Uma vez que o autómato B é acessível, existe uma palavra w tal que $p = \delta^\star{}_B(s_b, w)$.

Concluímos que $p = h(q)$, para o estado $q = \delta^\star{}_A(s_A, w)$. De facto, supondo que $h(q) \neq p$ e atendendo a que o autómato B é reduzido, existiria uma palavra x que distinguiria os estados $h(q)$ e p.

Assim, apenas uma das palavras $w_q x$ ou wx pertenceria a $\mathcal{L}(B)$. Uma vez que os autómatos A e B são equivalentes, concluiríamos que apenas uma das palavras $w_q x$ ou wx pertenceria a $\mathcal{L}(A)$ e chegaríamos a uma contradição, atendendo a que $\delta^\star{}_A(s_A, w_q x) = \delta^\star(q, x) = \delta^\star(s_A, wx)$.

3.4.4. Demonstre o Teorema 3.2.7.

Resolução. Sejam $[p] \neq [q]$ dois quaisquer estados do autómato quociente $A_{\equiv}$. Pela definição de classe de equivalência, os estados p e q pertencem a classes distintas, pelo que $p \not\equiv q$. Assim, existe uma palavra $w \in \Sigma^\star$ que os distingue, ou seja,

$$\delta^\star{}_A(p, w) \in F_A \text{ e } \delta^\star{}_A(q, w) \notin F_A$$

ou

$$\delta^\star{}_A(p, w) \notin F_A \text{ e } \delta^\star{}_A(q, w) \in F_A.$$

Para qualquer estado $p \in Q_A$ e palavra $w \in \Sigma^\star$, verifica-se que $\delta^\star{}_{\equiv}([p], w) = [\delta^\star{}_A(p, w)]$. Assim,

$$\delta^\star{}_{\equiv}([p], w) \in F_{\equiv} \text{ e } \delta^\star{}_{\equiv}([q], w) \notin F_{\equiv}$$

ou

$$\delta^\star{}_{\equiv}([p], w) \notin F_{\equiv} \text{ e } \delta^\star{}_{\equiv}([q], w) \in F_{\equiv}.$$

Concluímos que $[p] \not\equiv [q]$ e que todos os pares de estados do autómato $A_{\equiv}$ são distinguíveis, pelo que o autómato quociente é um autómato reduzido.

3.4.5. Para cada um dos seguintes autómatos, determine o autómato finito determinista mínimo equivalente:

(a)

δ	0	1
$\to p$	q	r
$*q$	p	r
$*r$	p	q

Solução.

δ_{min}	0	1
$\to p$	qr	qr
$*qr$	p	qr

(b)

δ	0	1
$\to *p$	p	q
q	q	r
$*r$	r	q

Solução.

δ_{min}	0	1
$\to *pr$	q	
q	q	pr

(c)

δ	0	1
$\to *p$	q	p
q	t	r
r	u	r
s	p	s
t	q	s
$*u$	t	u

Solução.

δ_{min}	0	1
$\rightarrow *pu$	qt	pu
qt	qt	rs
rs	pu	rs

(d)

δ	0	1
$\rightarrow *p$	s	q
q	t	r
$*r$	u	p
$*s$	p	t
t	q	u
$*u$	r	s

Solução.

δ_{min}	0	1
$\rightarrow *ps$	qs	qt
qt	qt	ru
$*ru$	ru	ps

(e)

δ	0	1
$\rightarrow *p$	q	s
q	t	r
$*r$	q	u
s	p	t
t	t	t
u	r	t

Solução.

δ_{min}	0	1
$\rightarrow *pr$	q	su
q	t	pr
su	pr	t
t	t	t

(f)

δ	0	1
$\rightarrow *p$	q	s
q	t	r
$*r$	q	u
s	p	t
t	p	t
u	r	t

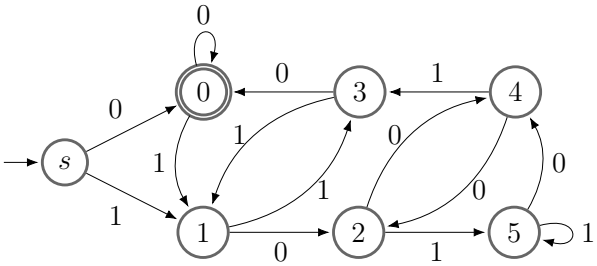
Solução.

δ_{min}	0	1
$\rightarrow *pr$	q	stu
q	stu	pr
stu	pr	stu

3.4.6. Considere a linguagem L das representações binárias de múltiplos de 6.

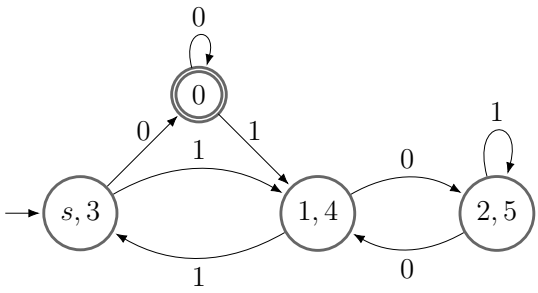
(a) Construa um AFD que reconheça a linguagem L.

Resolução. O AFD seguinte não reconhece a palavra vazia. Os múltiplos de 6 são as palavras binárias w, com valor decimal $d(w)$, tais que $d(w)$ mod $6 = 0$. Logo, existem 6 restos possíveis. Para definir as transições, aplicamos a propriedade: se $d(w)$ mod $6 = k$ então $d(w0)$ mod $6 = (2k)$ mod 6 e $d(w1)$ mod $6 = (2k + 1)$ mod 6.



(b) Mostre que o autómato obtido na alínea anterior é mínimo, ou então obtenha o AFD mínimo para L.

Resolução. Aplicando o algoritmo de identificação de pares de estados indistinguíveis ao autómato anterior, concluímos que os pares $\{s, 3\}$, $\{1, 4\}$ e $\{2, 5\}$ são indistinguíveis. Assim, o AFD mínimo é o seguinte:

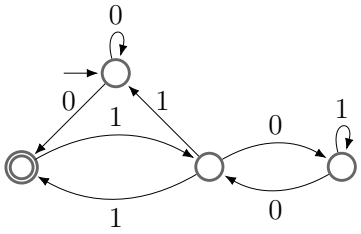


3.4.7. Considere a linguagem $L_1 = \{w \in \{0,1\}^\star : \ w^{-1} \notin L\}$, onde L é a linguagem definida no exercício anterior.

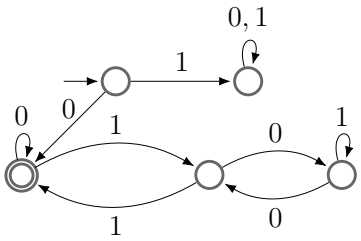
(a) Construa um AFD completo que reconheça a linguagem L_1.

Resolução. A linguagem L_1 é a linguagem complementar da linguagem reversa de L, ou seja, $L_1 = \{0,1\}^\star \setminus L^{-1}$.

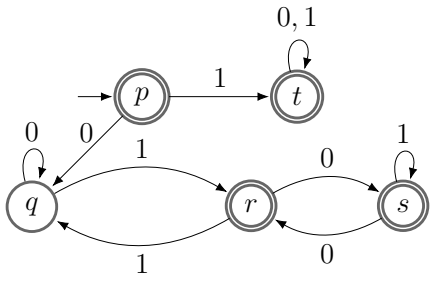
Partindo do AFD completo obtido na alínea (b) do exercício anterior, obtemos o seguinte AFND reconhecedor de L^{-1}:



Aplicando o algoritmo de determinização, obtemos o seguinte AFD completo equivalente:



Trocando os estados de aceitação com os de não aceitação, obtemos o seguinte AFD completo reconhecedor de L_1:



(b) Mostre que o autómato obtido na alínea anterior é mínimo, ou então obtenha o AFD mínimo de para L_1.

Resolução. Aplicando o algoritmo de identificação de pares de estados indistinguíveis, concluímos que os estados se distinguem uns dos outros, pelo que o autómato obtido na alínea anterior é já o AFD mínimo reconhecedor de L_1. Na tabela seguinte indicamos palavras que distinguem os pares de estados.

	q	ε			
$*$	r	0	ε		
$*$	s	0	ε	1	
$*$	t	0	ε	1	01
		p	q	r	s
		$*$		$*$	$*$

3.4.8. Considere as linguagens

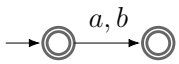
$$L_1 = \{w \in \{a,b\}^\star : \quad \text{o tamanho de } w \text{ não é superior a } 1\}$$

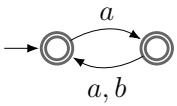
e

$$L_2 = \{w \in \{a,b\}^\star : \quad \text{em qualquer posição ímpar de } w \text{ o símbolo é } a\}.$$

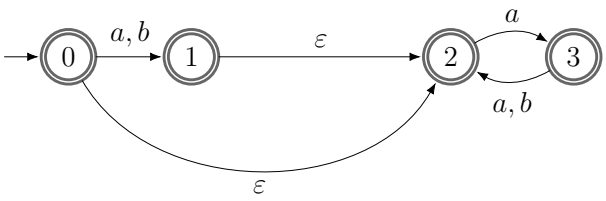
(a) Construa um AFNDε que reconheça a linguagem L_1L_2.

Resolução. Começamos por construir AFD reconhecedores de L_1 e de L_2:





Em seguida, construímos um AFNDε para L_1L_2 ligando, com transições ε, os estados de aceitação do primeiro autómato com o estado inicial do segundo autómato:



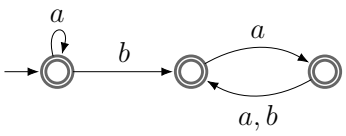
(b) Determinize o autómato que obteve na alínea anterior.

Resolução. Aplicando o algoritmo de determinização para AFNDε obtemos o AFD com a seguinte tabela de transições:

δ_D	a	b
$\rightarrow *\{0,2\}$	$\{1,2,3\}$	$\{1,2\}$
$*\{1,2,3\}$	$\{2,3\}$	$\{2\}$
$*\{1,2\}$	$\{3\}$	$\emptyset$
$*\{2,3\}$	$\{2,3\}$	$\{2\}$
$*\{2\}$	$\{3\}$	$\emptyset$
$*\{3\}$	$\{2\}$	$\{2\}$

(c) Construa o AFD mínimo reconhecedor da linguagem L_1L_2.

Resolução. Aplicando o algoritmo de identificação de pares de estados indistinguíveis, obtemos o AFD mínimo:



3.4.9. Considere as linguagens

$$L_1 = \{w \in \{a,b\}^\star : \ w \text{ começa com } a \text{ e tem tamanho par}\}$$

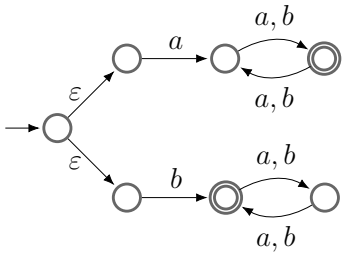
e

$$L_2 = \{w \in \{a,b\}^\star : \ w \text{ começa com } b \text{ e tem tamanho ímpar}\}.$$

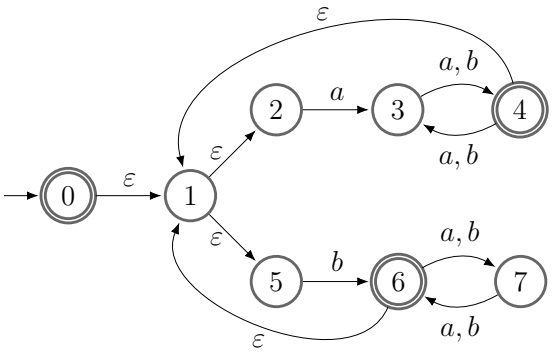
(a) Construa um AFNDε que reconheça a linguagem $L = (L_1 \cup L_2)^\star$.

Resolução. Começamos por construir AFD reconhecedores de L_1 e de L_2:

Em seguida, construímos um AFNDε para $L_1 \cup L_2$ introduzindo um novo estado inicial ligado por transições ε aos estados iniciais dos autómatos (os quais deixam de ser iniciais):



Finalmente construímos um AFNDε para $(L_1 \cup L_2)^\star$. Começamos por ligar os estados de aceitação do autómato anterior ao seu estado inicial por transições ε; em seguida, introduzimos um novo estado inicial, de aceitação (para que o autómato reconheça a palavra vazia), ligado por uma transição ε ao estado inicial do autómato anterior (o qual deixa de ser inicial):



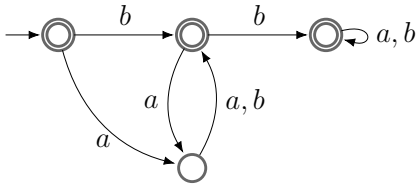
(b) Determinize o autómato que obteve na alínea anterior.

Resolução. Aplicando o algoritmo de determinização para AFNDε obtemos o AFD com a seguinte tabela de transições:

δ_D	a	b
$\to *\{0,1,2,5\}$	$\{3\}$	$\{1,2,5,6\}$
$\{3\}$	$\{1,2,4,5\}$	$\{1,2,4,5\}$
$*\{1,2,5,6\}$	$\{3,7\}$	$\{1,2,5,6,7\}$
$*\{1,2,4,5\}$	$\{3\}$	$\{1,2,3,5,6\}$
$\{3,7\}$	$\{1,2,4,5,6\}$	$\{1,2,4,5,6\}$
$*\{1,2,5,6,7\}$	$\{1,2,3,5,6,7\}$	$\{1,2,5,6,7\}$
$*\{1,2,3,5,6\}$	$\{1,2,3,4,5,7\}$	$\{1,2,4,5,6,7\}$
$*\{1,2,4,5,6\}$	$\{3,7\}$	$\{1,2,3,5,6,7\}$
$*\{1,2,3,5,6,7\}$	$\{1,2,3,4,5,6,7\}$	$\{1,2,4,5,6,7\}$
$*\{1,2,3,4,5,7\}$	$\{1,2,3,4,5,6\}$	$\{1,2,3,4,5,6\}$
$*\{1,2,4,5,6,7\}$	$\{1,2,3,5,6,7\}$	$\{1,2,3,5,6,7\}$
$*\{1,2,3,4,5,6,7\}$	$\{1,2,3,4,5,6,7\}$	$\{1,2,3,4,5,6,7\}$
$*\{1,2,3,4,5,6\}$	$\{1,2,3,4,5,7\}$	$\{1,2,3,4,5,6,7\}$

(c) Construa o AFD mínimo reconhecedor da linguagem L.

Resolução. Aplicando o algoritmo de identificação de pares de estados indistinguíveis, obtemos o AFD mínimo:



3.4.10. Aplique o Algoritmo 3.3.1 para testar a equivalência dos autómatos ilustrados na Figura 3.1.1.

Resolução. Aplicando o algoritmo de identificação de pares de estados indistinguíveis aos dois autómatos em simultâneo, obtemos:

$*\ q_1$	ε					
q_2	b	ε				
$\to p_0$		ε	b			
$*\ p_1$	ε		ε	ε		
$*\ p_2$	ε		ε	ε		
p_3	b	ε		b	ε	ε
	q_0	q_1	q_2	p_0	p_1	p_2
	$\uparrow$	$*$			$*$	$*$

Uma vez que os estados iniciais q_0 e p_0 são indistinguíveis, os autómatos são equivalentes.

4 Expressões regulares

4.6 Exercícios

4.6.1. Construa expressões regulares para as linguagens referidas nos exercícios 2.6.2, 2.6.3 e 2.6.10.

Solução.

Linguagens do exercício 2.6.2.

(a) $a + b$

(b) ε

(c) $a(a + b)^*$

(d) $b(a + b)^*$

(e) $ab(a + b)^*$

(f) $(a + b)^*a$

(g) $(a + b)^*b$

(h) $a(a + b)^*a$

(i) $a(a + b)^*aa(a + b)^*a$

(j) $a(a + b)^*a + b(a + b)^*b$

(k) b^*ab^*

(l) $(a + b)^*a(a + b)^*a$

(m) $b^*(\varepsilon + a)b^*$

Linguagens do exercício 2.6.3.

(a) a^*b

(b) a^*b^*

(c) aa^*bb^*

(d) $(aa)^*$

(e) $a(aa)^*$

(f) $(aa)^*(bb)^*$

(g) $((a+b)(a+b))^*$

(h) $((a+b)(a+b)(a+b))^*$

(i) $((a+b)(a+b)(a+b))^*(\varepsilon+a+b)$

(j) $(b^*ab^*ab^*a)^*\, b^*$

(k) $(b^*aa^*bb^*aa^*b)^*\, b^*$

(l) $(ab+b)^*(a+\varepsilon)$

(m) $(aab+ab+b)^*(aa+a+\varepsilon)$

(n) $(ab)^*(\varepsilon+a)+(ba)^*(\varepsilon+b)$

(o) $((a+aa)(b+bb))^*\,(\varepsilon+a+aa)+$
$((b+bb)(a+aa))^*\,(\varepsilon+b+bb)$

(p) $\varepsilon+a(a+bb^*a)^*+b(b+aa^*b)^*$

Linguagens do exercício 2.6.10.

(a) a^*b^*

(b) $a^*b^*c^*$

(c) $(aa)^*(bb)^*$

(d) $a(aa)^*b(bb)^*$

(e) $a(a+b)^*b$

(f) $(0+1)^*0$

(g) $(0+1)^*01$

(h) $(0+1)^*(0+11)$

(i) $0(0+1)(0+1)^*$

(j) $(0+1)^*0(0+1)$

(k) $010(0+1)^*+(0+1)^*110$

(l) $010(0+1)^*110$

4.6.2. 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.

Resolução. A afirmação é verdadeira. Se R_i é uma expressão regular associada à linguagem regular L_i, $i = 1,\ldots n$, então $\sum_{i=1}^{n} R_i = R_1 + R_2 + \cdots + R_n$ é uma expressão regular associada à linguagem $L = \bigcup_{i=1}^{n} L_i$.

(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.

Resolução. A afirmação é verdadeira. Pelo teorema de Thompson, podemos construir um AFNDε A_i que reconhece a linguagem regular L_i associada à expressão regular R_i, $i = 1,\ldots n$. Cada um destes autómatos pode ser convertido num AFD completo equivalente. Podemos aplicar repetidamente, a dois autómatos de cada vez, a construção do autómato produto para a intersecção de linguagens obtendo um AFD reconhecedor da linguagem $L = \bigcap_{i=1}^{n} L_i$. Finalmente, aplicamos o Teorema de Kleene a este autómato

para obter uma expressão regular associada à linguagem L.

(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.

Resolução. A afirmação é falsa. Começamos por salientar que o método indicado na alínea (a) não é válido para uma união infinita de linguagens pois a sua aplicação daria origem a uma expressão regular infinita.

Como contra-exemplo à afirmação, consideremos a linguagem não regular $L = \{a^n b^n : \ n > 0\}$. Temos que $L = \bigcup_{n=1}^{\infty} L_n$, onde $L_n = \{a^n b^n\}$. Cada uma das linguagens L_n é regular pois tem uma única palavra ($a^n b^n$ é a expressão regular associada a L_n).

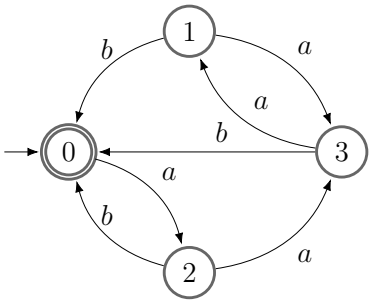
(d) Se $\forall n \in \mathbb{N}$, L_n é uma linguagem regular então $L = \bigcap_{n=1}^{\infty} L_n$ é uma linguagem regular.

Resolução. A afirmação é falsa. Começamos por notar que o método indicado na alínea (b) não é válido para uma intersecção infinita de linguagens pois a sua aplicação daria origem a um autómato com um número infinito de estados.

Como contra-exemplo à afirmação, consideremos a linguagem não regular $L = \{a^n b^n : \ n > 0\}$ e a decomposição $L = \bigcup_{n=1}^{\infty} L_n$, onde $L_n = \{a^n b^n\}$ é regular.

Como a classe das linguagens regulares é fechada para a operação complementar, concluímos que cada uma linguagens $\overline{L_n}$ é regular, enquanto $\overline{L} = \bigcap_{n=1}^{\infty} \overline{L_n}$ não é regular (caso contrário L também seria regular).

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



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

Resolução. A linguagem $\mathcal{L}(A)$ é constituída pela palavra vazia e por todas as palavras sobre o alfabeto $\Sigma = \{a, b\}$ que começam com a, terminam em b e não têm dois b´s consecutivos.

(b) Usando o método de identificação de estados indistinguíveis, determine se o autómato A é ou não mínimo. Indique quantas iterações realizou e, para

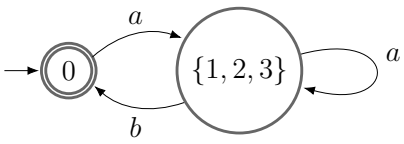
cada par de estados distinguíveis, apresente uma palavra que os distinga.

Resolução. Uma vez que o autómato não é completo, temos de o completar introduzindo um estado ratoeira (r). Aplicando o método de eliminação de estados, obtemos o quadro abaixo indicado ao fim de 3 iterações: na primeira, marcamos os pares aceitação/rejeição que são distinguidos pela palavra vazia; na segunda, distinguimos apenas os estados q_1, q_2 e q_3 do estado r pelo símbolo b; na terceira, não fazemos qualquer alteração ao quadro e o algoritmo termina.

q_1	ε			
q_2	ε			
q_3	ε			
r	ε	b	b	b
	q_0	q_1	q_2	q_3
	$*$			

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

Resolução. Pela alínea anterior, os pares $\{q_1, q_2\}$, $\{q_1, q_3\}$ e $\{q_2, q_3\}$ são indistinguíveis. No autómato quociente vão fazer parte da mesma classe de equivalência. Assim, o autómato mínimo é determinado pelo seguinte diagrama de transições (o estado ratoeira não está representado):



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

Resolução. Uma possível expressão regular para $\mathcal{L}(A)$ é $R = (aa^*b)^*$.

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

Resolução. Uma expressão regular para $\mathcal{L}(A)^{-1}$, construída a partir da expressão indicada na alínea anterior, é:

$$S = \text{REV}(R) = \text{REV}((aa^*b)^*) = (\text{REV}(aa^*b))^* = (b\,\text{REV}(aa^*))^* = (ba^*a)^*.$$

4.6.4. Demonstre o Lema 4.3.13.

Resolução.

Caso base. Se $R = \varepsilon$ então

$$\mathcal{L}(h(\varepsilon)) = \mathcal{L}(\varepsilon) = \{\varepsilon\} = h(\{\varepsilon\}) = h(\mathcal{L}(\varepsilon)).$$

Se $R = a$ para $a \in \Sigma$ então $h(a) = w$ para alguma palavra $w \in \Gamma^\star$. Assim,

$$\mathcal{L}(h(a)) = \mathcal{L}(w) = \{w\} = h(\{a\}) = h(\mathcal{L}(a)).$$

Passo indutivo. Consideremos como hipótese de indução que a propriedade é válida para as expressões regulares S e T.

Se $R = S + T$ então

$$\begin{aligned}\mathcal{L}(h(S+T)) &= \mathcal{L}(h(S) + h(t)) = \mathcal{L}(h(S)) \cup \mathcal{L}(h(T)) \\ &= h(\mathcal{L}(S)) \cup h(\mathcal{L}(T)) = h(\mathcal{L}(S) \cup \mathcal{L}(T)) = h(\mathcal{L}(S+T)).\end{aligned}$$

Se $R = ST$ então

$$\begin{aligned}\mathcal{L}(h(ST)) &= \mathcal{L}(h(S)h(t)) = \mathcal{L}(h(S))\mathcal{L}(h(T)) \\ &= h(\mathcal{L}(S))h(\mathcal{L}(T)) = h(\mathcal{L}(S)\mathcal{L}(T)) = h(\mathcal{L}(ST)).\end{aligned}$$

Se $R = S^*$ então

$$\begin{aligned}\mathcal{L}(h(S^*)) &= \mathcal{L}(h(S)^*) = \mathcal{L}(h(S))^\star \\ &= h(\mathcal{L}(S))^\star = h(\mathcal{L}(S)^\star) = h(\mathcal{L}(S^*)).\end{aligned}$$

4.6.5. 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.

Resolução. Falsa. Por exemplo, $L = \{a^n b^n : n \in \mathbb{N}_0\}$ e $h(a) = h(b) = a$.

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

Resolução. Verdadeira. Ver o Teorema 4.3.14.

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

Resolução. Falsa. Por exemplo, $L = \{a^n b^n : n \in \mathbb{N}_0\}$ e $h(a) = h(b) = a$.

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

Resolução. Verdadeira. Ver o Teorema 4.3.15.

4.6.6. 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 $h(\mathcal{L}(S))$.

Resolução. $h(S) = (00)^*1 + 1(00)^*$ especifica a linguagem $h(\mathcal{L}(S)) = \mathcal{L}(h(S))$.

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

Resolução. $\mathcal{L}(S)$ é constituída pelas palavras começadas por uma sequência, eventualmente nula, de a´s e terminadas em b e pelas palavras que começam em b seguidas por uma sequência, eventualmente nula, de a´s.

$h(\mathcal{L}(S))$ é constituída pelas palavras que começam com um número par de 0´s e que terminam em 1 e pelas palavras começadas por 1, seguidas de um número par de 0´s.

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

Resolução. L_1 é a linguagem associada à expressão regular $(ab)^*$ e L_2 é a linguagem associada à expressão regular $\emptyset$.

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

Resolução. A linguagem associada à expressão regular $R + S$ é $\mathcal{L}(R + S) = \mathcal{L}(R) \cup \mathcal{L}(S)$ e a linguagem associada à expressão regular S é $\mathcal{L}(S)$. Uma vez que $\mathcal{L}(R) \subseteq \mathcal{L}(S)$, $\mathcal{L}(R) \cup \mathcal{L}(S) = \mathcal{L}(S)$. Logo, pela definição de igualdade de expressões regulares, $R + S = S$.

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

Resolução. Vamos mostrar por indução sobre $n \in \mathbb{N}_0$ que $\mathcal{L}(AB^n) \subseteq \mathcal{L}(X)$.

Para $n = 0$ temos que $\mathcal{L}(A) \subseteq \mathcal{L}(X)$ uma vez $\mathcal{L}(A + XB) \subseteq \mathcal{L}(X)$.

Consideremos como hipótese de indução que $\mathcal{L}(AB^n) \subseteq \mathcal{L}(X)$. Então

$$\mathcal{L}\big(AB^{n+1}\big) = \mathcal{L}(AB^nB) \subseteq \mathcal{L}(XB).$$

Logo, $\mathcal{L}(A + AB^{n+1}) \subseteq \mathcal{L}(A + XB) \subseteq \mathcal{L}(X)$ e também $\mathcal{L}(AB^{n+1}) \subseteq \mathcal{L}(X)$.

Concluímos que $\mathcal{L}(AB^*) = \bigcup_{n=0}^{\infty} AB^n \subseteq \mathcal{L}(X)$.

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

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

Resolução.

(A) Seja $w \in \mathcal{L}(R^*R^*) = \mathcal{L}(R^*)\mathcal{L}(R^*)$. Então existem palavras $x, y \in \mathcal{L}(R^*)$ tais que $w = xy$. Existem $m, n \in \mathbb{N}_0$ tais que $x \in \mathcal{L}(R)^m$ e $y \in \mathcal{L}(R)^n$. Assim $w = xy \in \mathcal{L}(R)^m\mathcal{L}(R)^n = \mathcal{L}(R)^{m+n} \subseteq \mathcal{L}(R^*)$.

(B) Uma vez que $\mathcal{L}(\varepsilon) \subseteq \mathcal{L}(R^*)$, temos que

$$\mathcal{L}(R^*) = \mathcal{L}(\varepsilon R^*) \subseteq \mathcal{L}(\varepsilon)\mathcal{L}(R^*) \subseteq \mathcal{L}(R^*)\mathcal{L}(R^*) = \mathcal{L}(R^*R^*).$$

De (A) e (B) concluímos que $\mathcal{L}(R^*R^*) = \mathcal{L}(R^*)$.

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

Resolução. $R^*S + S = (R^* + \varepsilon)S = R^*S$.

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

Resolução. $R + SS^*R = (\varepsilon + SS^*)R = S^*R$.

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

Resolução. Temos que

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

De igual modo,

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

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

Resolução.

(A) Temos que $\mathcal{L}(R) = \mathcal{L}(R\varepsilon) = \mathcal{L}(R)\mathcal{L}(\varepsilon) \subseteq \mathcal{L}(R^*)\mathcal{L}(S^*) = \mathcal{L}(R^*S^*)$ e que $\mathcal{L}(S) = \mathcal{L}(\varepsilon S) = \mathcal{L}(\varepsilon)\mathcal{L}(S) \subseteq \mathcal{L}(R^*)\mathcal{L}(S^*) = \mathcal{L}(R^*S^*)$. Assim,

$$\mathcal{L}(R + S) = \mathcal{L}(R) \cup \mathcal{L}(S) \subseteq \mathcal{L}(R^*S^*).$$

Logo, $\mathcal{L}((R + S)^*) \subseteq \mathcal{L}((R^*S^*)^*)$.

(B) Temos que $\mathcal{L}(R) \subseteq \mathcal{L}(R + S)$ e portanto $\mathcal{L}(R^*) \subseteq \mathcal{L}((R + S)^*)$. De igual modo, temos que $\mathcal{L}(S) \subseteq \mathcal{L}(R + S)$ e portanto

$$\mathcal{L}(S^*) \subseteq \mathcal{L}((R + S)^*).$$

Assim,

$$\mathcal{L}(R^*S^*) = \mathcal{L}(R^*)\mathcal{L}(S^*) \subseteq \mathcal{L}((R + S)^*)\mathcal{L}((R + S)^*) = \mathcal{L}((R + S)^*).$$

De (A) e (B) concluímos que $\mathcal{L}((R + S)^*) = \mathcal{L}((R^*S^*)^*)$.

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

Resolução. Usando a alínea anterior, temos que

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

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

Resolução. Temos que $(\varepsilon + R)^+ = (\varepsilon + R)(\varepsilon + R)^*$. Pela alínea anterior, $(\varepsilon + R)^* = R^*$. Assim, $(\varepsilon + R)^* = (\varepsilon + R)R^*$ e pela alínea (d), concluímos que $(\varepsilon + R)^* = R^*$.

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

Resolução. Começamos por provar por indução que $R(SR)^n = (RS)^n R$, qualquer que seja $n \in \mathbb{N}_0$. De facto, temos que $R(SR)^0 = R\varepsilon = R = \varepsilon R = (RS)^0 R$, $R(SR)^1 = R(SR) = (RS)R = (RS)^1 R$. Admitindo, por hipótese que $R(SR)^n = (RS)^n R$ vem que

$$\begin{aligned} R(SR)^{n+1} &= R((SR)^n(SR)) = (R(SR)^n)(SR) \\ &= ((RS)^n R)(SR) = ((RS)^n(RS))R = (RS)^{n+1}R. \end{aligned}$$

Temos ainda que,

$$\begin{aligned} \mathcal{L}(R(SR)^*) &= \mathcal{L}(R)\left(\bigcup_{n=0}^{\infty} \mathcal{L}(SR)^n\right) = \bigcup_{n=0}^{\infty} \mathcal{L}(R)\mathcal{L}(SR)^n = \bigcup_{n=0}^{\infty} \mathcal{L}(R(SR)^n) \\ &= \bigcup_{n=0}^{\infty} \mathcal{L}((RS)^n R) = \bigcup_{n=0}^{\infty} \mathcal{L}(RS)^n \mathcal{L}(R) \\ &= \left(\bigcup_{n=0}^{\infty} \mathcal{L}(RS)^n\right) \mathcal{L}(R) = \mathcal{L}((RS)^*)\mathcal{L}(R) = \mathcal{L}((RS)^* R). \end{aligned}$$

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

Resolução.

$\subseteq$. Temos que $\mathcal{L}(\varepsilon) \subseteq \mathcal{L}((R^*S)^*R^*)$, $\mathcal{L}((R^*S)^*R^*R) \subseteq \mathcal{L}((R^*S)^*R^*)$ e

$$\mathcal{L}((R^*S)^*R^*S) \subseteq \mathcal{L}((R^*S)^*) \subseteq \mathcal{L}((R^*S)^*R^*).$$

Assim, $\mathcal{L}(\varepsilon + (R^*S)^*R^*R + (R^*S)^*R^*S) \subseteq \mathcal{L}((R^*S)^*R^*)$, ou seja,

$$\mathcal{L}(\varepsilon + (R^*S)^*R^*(R + S)) \subseteq \mathcal{L}((R^*S)^*R^*).$$

Aplicando agora a propriedade indicada no Exercício 4.6.8 com $A = \varepsilon$, $B = (R + S)$ e $X = (R^*S)^*R^*$, concluímos que

$$\mathcal{L}((R + S)^*) \subseteq \mathcal{L}((R^*S)^*R^*).$$

$\supseteq$. Atendendo a que $\mathcal{L}(R) \subseteq \mathcal{L}(R+S)$ e $\mathcal{L}(S) \subseteq \mathcal{L}(R+S)$, podemos afirmar que

$$\mathcal{L}((R^*S)^*R^*) \subseteq \mathcal{L}(((R+S)^*(R+S))^*(R+S)^*) \subseteq \mathcal{L}((R+S)^*).$$

Concluímos que $\mathcal{L}((R+S)^*) = \mathcal{L}((R^*S)^*R^*)$ e por aplicação directa da propriedade da alínea anterior, concluímos que $\mathcal{L}((R^*S)^*R^*) = \mathcal{L}(S^*R)^*S^*)$.

4.6.10. 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.

Resolução. Seja $A = (Q_A, \Sigma, \delta, s_A, F_A)$ um AFD reconhecedor de L. A partir de de A definimos o AFNDε $B = (Q_B, \Sigma, \delta_B, s_B, F_B)$, reconhecedor de L_1, onde:

- $Q_B = Q_A \times Q_A \times Q_A \cup \{s_B\}$;
- $F_B = \{(q, q, f) : \ q \in Q_A \text{ e } f \in F_A\}$;
- $\delta_B(s_B, \varepsilon) = \{(s, q, q) : \ q \in Q_A\}$;
- $\delta_B\left((p, q, r), a\right) = \{(\delta(p, a), q, \delta(r, a))\}$.

Este autómato simula o funcionamento em paralelo de duas cópias do autómato A: a primeira cópia, A_1, lê a primeira metade da palavra ww começando no estado s_A; a segunda cópia, A_2, lê a segunda metade de ww começando num certo estado q.

A partir do novo estado inicial, s_B, o autómato B usa o não determinismo (transições ε) para "adivinhar" o estado intermédio $q = \delta^\star{}_A(s, w)$ sem ter que ler a primeira parte de ww.

O autómato B atinge um estado (p, q, r) por leitura de uma palavra w se A_1 atingir o estado p partindo de s_A e A_2 atingir o estado r partindo do estado q.

Assim, o autómato B reconhece uma palavra w se e só só o autómato A_1, partindo de s, atinge um certo estado q enquanto o autómato A_2, partindo de q, atinge um estado de aceitação f.

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

Resolução. A linguagem $L = \{a^n b : \ n \geq 0\}$, definida sobre o alfabeto $\Sigma = \{a, b\}$, é regular uma vez que é especificada pela expressão regular a^*b. No entanto $L_2 = \{ww : \ w \in L\} = \{a^n b a^n b : \ n \geq 0\}$ não é regular.

Vamos admitir que L_2 é regular. Pelo lema da bombagem, Lema 4.4.1, existe um inteiro positivo n tal que qualquer palavra $w \in L$ de tamanho superior a n satisfaz as condições (1) a (4) indicadas no lema.

Consideremos a palavra $w = a^n b a^n b \in L_2$ que tem tamanho $2n + 2 > n$. Dada uma qualquer partição $w = xyz$, uma vez que $|xy| < n$, tanto x como y são sequências de a´s. Isto é $x = a^p$, $y = a^q$, $z = a^{n-p-q} b a^n b$, com $0 < p + q < n$. Pela condição (4), para $k = 0$, a palavra $xz \in L_2$. Mas $xz = a^{n-q} b a^n b \notin L_2$ uma vez que $q = |y| > 1$ e portanto $n - q \neq n$. Contradição! Logo, L_2 não é regular.

4.6.11. 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.

Resolução.

($\Leftarrow$) Seja $A = (Q, \Sigma, \delta, s, \{f\})$ um AFD com um único estado de aceitação em que as transições a partir do estado de aceitação são para um estado ratoeira $r \in Q$.

Sejam x e y duas quaisquer palavras tais que $x \in \mathcal{L}(A)$ e $y \neq \varepsilon$. Então $\delta^\star(s, xy) = \delta^\star(\delta^\star(s, x), y) = \delta^\star(f, y) = r$. Logo, $xy \notin \mathcal{L}(A)$. Concluímos assim que $\mathcal{L}(A)$ é livre de prefixos.

($\Rightarrow$) Consideremos um AFD completo reconhecedor de uma linguagem regular livre de prefixos com vários estados de aceitação. As transições a partir dos estados de aceitação são necessariamente para estados ratoeira, ou seja, as linguagens direitas dos estados de aceitação são iguais à linguagem vazia, caso contrário, a linguagem não seria livre de prefixos. Logo, os estados de aceitação são indistinguíveis e no autómato mínimo reduzem-se a um único estado de aceitação.

(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.

Resolução. Se $R = \varepsilon$ então $R^* = \varepsilon$ e a linguagem associada a R^* é reconhecida por um AFD com um único estado de aceitação. Vamos assim considerar que $R \neq \varepsilon$.

Pela alínea anterior, o AFD mínimo que reconhece $\mathcal{L}(R)$ tem um único estado de aceitação com transições para um estado ratoeira. Para além disso, o estado inicial é diferente do estado de aceitação uma vez que ε não faz parte de $\mathcal{L}(R)$. Podemos eliminar essas transições para o estado ratoeira e fundir o estado de aceitação com o estado inicial (o estado inicial passa também a ser de aceitação). Este novo AFD reconhece a linguagem $\mathcal{L}(R^*)$. Notamos que a linguagem $\mathcal{L}(R^*)$ não é livre de prefixos, pois ε é um prefixo próprio das palavras de $\mathcal{L}(R)$.

Mais precisamente, a partir do AFD mínimo reconhecedor da linguagem $\mathcal{L}(R)$, $A = (Q, \Sigma, \delta_A, s, \{f\})$, construímos o AFD reconhecedor de $\mathcal{L}(R^*)$, $B = (Q \setminus \{f\}, \Sigma, \delta_B, s, \{s\})$, onde a função de transição δ_B é definida para $q \in Q \setminus \{f\}$ e $a \in \Sigma$ por:

- se $\delta_A(q, a) = f$ então $\delta_B(q, a) = s$;
- caso contrário, $\delta_B(q, a) = \delta_A(q, a)$.

(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.

Resolução. Seja $A = (Q, \Sigma, \delta_A, s, \{s\})$ um AFD em que o estado de aceitação é o estado inicial.

Consideremos o AFD $B = (Q \cup \{f, r\}, \Sigma, \delta_B, s, \{f\})$ em que a função de transição δ_B é definida para $a \in \Sigma$, por:

- para $q \in Q$, se $\delta_A(q, a) = s$ então $\delta_B(q, a) = f$;
- para $q \in Q$, se $\delta_A(q, a) \neq s$ então $\delta_B(q, a) = \delta_A(q, a)$;
- $\delta_B(f, a) = \delta_B(r, a) = r$.

Pela alínea (a), a linguagem $\mathcal{L}(B)$ é livre de prefixos. Seja R uma expressão regular tal que $\mathcal{L}(B) = \mathcal{L}(R)$.

O reconhecimento de uma palavra não vazia pelo autómato A é obtido percorrendo um ou mais ciclos que começam e terminam em s, os quais correspondem a caminhos de s para f no autómato B. Afirmamos que $\mathcal{L}(A) = \mathcal{L}(B)^\star = \mathcal{L}(R^*)$, deixando ao cuidado do leitor os detalhes da demonstração.

(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.

Resolução.

($\Rightarrow$) Seja $A = (Q, \Sigma, \delta_A, s, \{f\})$ um AFD completo, com um único estado de aceitação que reconhece L. A partir de A definimos o AFD $B = (Q \cup \{r\}, \Sigma, \delta_B, s, \{f\})$, que difere do autómato A apenas nas transições a partir do estado de aceitação que passam a ser para um novo estado ratoeira r. Mais precisamente: para $a \in \Sigma$ e $q \in Q \setminus \{f\}$, $\delta_B(q, a) = \delta_A(q, a)$; para $a \in \Sigma$ e $q \in \{f, r\}$, $\delta_B(q, a) = r$. Assim, pela alínea (a), concluímos que linguagem $\mathcal{L}(B) \subseteq L$ é livre de prefixos e podemos associar-lhe uma expressão regular S.

Partindo de A, definimos ainda o AFD $C = (Q, \Sigma, \delta_A, f, \{f\})$, semelhante ao autómato A excepto que o estado inicial de C é estado de aceitação de A. Pela

alínea anterior, a linguagem reconhecida pelo AFD C é $\mathcal{L}(R^*)$ para alguma expressão regular R associada a uma linguagem regular livre de prefixos.

Podemos particionar qualquer palavra $w \in L$ em $w = xy$, onde:

(i) $x \in L$ e nenhum prefixo próprio de x pertence a L, isto é, $\delta_A(s, x) = f$ e se $x = uv$ com $v \neq \varepsilon$ então $\delta_A(s, u) \neq f$. Estas palavras são precisamente as reconhecidas pelo AFD B;

(ii) y é tal que $\delta_A(f, y) = f$. Estas palavras são precisamente as reconhecidas pelo AFD C.

Concluímos assim que $L = \mathcal{L}(B)\mathcal{L}(C) = \mathcal{L}(SR^*)$.

($\Leftarrow$) Sejam S e R expressõs regulares associadas às linguagens livres de prefixos, $\mathcal{L}(R)$ e $\mathcal{L}(S)$. Pela alínea (a), existe um AFD $A = (Q_A, \Sigma, \delta_A, s_A, \{f_A\})$, com um único estado de aceitação, distinto do estado inicial, que reconhece $\mathcal{L}(S)$. Para além disso, as transições a partir do estado de aceitação f_A são para um estado ratoeira, o qual pode ser eliminado.

Pela alínea (b), existe um AFD $B = (Q_B, \Sigma, \delta_B, s_B, \{s_B\})$, com um único estado de aceitação que é também o estado inicial e que reconhece $\mathcal{L}(R^*)$.

Assim, fundindo o estado de aceitação do autómato A com o estado inicial do autómato B, obtemos um AFD C reconhecedor de $\mathcal{L}(SR^*)$:

$$C = (Q_A \setminus \{f_A\} \cup Q_B, \Sigma, \delta_C, s_A, \{s_B\}),$$

onde a função de transição δ_C é definida por: para $q \in Q_A$ e $a \in \Sigma$, se $\delta_A(q, a) \neq f_A$ então $\delta_C(q, a) = \delta_A(q, a)$ e se $\delta_A(q, a) = f_A$ então $\delta_C(q, a) = s_B$; para $q \in Q_B$ e $a \in \Sigma$, $\delta_C(q, a) = \delta_B(q, a)$.

4.6.12. 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.

Resolução. Seja $A = (Q, \Sigma, \delta, s, F)$ um AFD reconhecedor de uma linguagem regular $\mathcal{L}(A)$. Para $f \in F$ definimos o AFD com um único estado de aceitação $A = (Q, \Sigma, \delta, s, f)$. Assim, $\mathcal{L}(A) = \bigcup_{f \in F} \mathcal{L}(A_f)$.

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

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

Resolução. Seja $A = (Q, \Sigma, \delta, s, F)$ um AFD reconhecedor de L_1. Para cada estado $q \in Q$ determinamos se a sua linguagem direita contém alguma palavra da linguagem L_2, isto é, se $\mathcal{L}_{\text{dir}}(q) \cap L_2 \neq \emptyset$. Para tal:

(a) construímos o autómato produto para a intersecção do AFD reconhecedor de $\mathcal{L}_{\text{dir}}(q)$, $A_q = (Q, \Sigma, \delta, q, F)$ com um AFD B reconhecedor de L_2;

(b) verificamos se os estados de aceitação no autómato produto são atingíveis a partir do estado inicial.

Seja então $F' = \{q \in Q : \ \mathcal{L}_{\text{dir}}(q) \cap L_2 \neq \emptyset\}$. Deixamos ao cuidado do leitor mostrar que o autómato $A = (Q, \Sigma, \delta, s, F')$ reconhece a linguagem $L_1 \,/\, L_2$.

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

Resolução. Se uma linguagem é finita então basta fixar n maior que o tamanho da maior palavra da linguagem.

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

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

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

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

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

Resolução. A classe das linguagens regulares é fechada para o complementar. Assim, se $L = \{w \in \{a, b\}^\star : \ w \neq w^{-1}\}$ fosse regular então também $\overline{L} = \{w \in \{a, b\}^\star : \ w = w^{-1}\}$ seria regular, o que é falso (confirmar com a alínea anterior).

(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 $k = 0$.

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

Resolução. Que a relação $\sim_A$ é de equivalência decorre das propriedades reflexiva, simétrica e transitiva da relação de igualdade entre estados do autómato.

Se $x \sim_A y$ então pela definição de $\sim_A$, ${\delta^\star}_A(s,x) = {\delta^\star}_A(s,y)$. Logo, ${\delta^\star}_A(s,xa) = \delta_A({\delta^\star}_A(s,x),a) = \delta_A({\delta^\star}_A(s,y),a) = \delta_A(s,xa)$., pelo que $xa \sim_A ya$ e portanto $\sim_A$ é uma congruência à direita.

Se $x \sim_A y$ então $x \in L \iff {\delta^\star}_A(s,x) \in F \iff {\delta^\star}_A(s,y) \in F \iff y \in L$. Logo, a relação $\sim_A$ refina L.

A cada palavra x corresponde um único estado do autómato $q = \delta^\star(s,x)$. Assim, as classes de equivalência da relação $\sim_A$ são as linguagens esquerdas dos estados do autómato, $\mathcal{L}_{\text{esq}}(q) = \{x \in \Sigma^* : \ {\delta^\star}_A(s,x) = q\}$. Uma vez que o número de estados do autómato é finito, concluímos que a relação $\sim_A$ tem índice finito.

4.6.17. Demonstre a propriedade (4.5.3).

Resolução. Para $y = \varepsilon$, temos que $\delta^\star([x],y) = \delta^\star([x],\varepsilon) = [x] = [x\varepsilon] = [xy]$.

Seja agora $y = za$ com $a \in \Sigma$ e $z \in \Sigma^\star$. Como hipótese de indução, consideremos que $\delta^\star([x],z) = [xz]$. Temos que $\delta^\star([x],y) = \delta^\star([x],za) = \delta(\delta^\star([x],z),a) = \delta([xz],a) = [xza] = [xy]$.

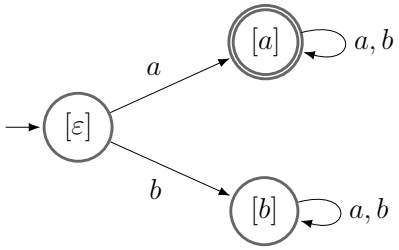
4.6.18. Seja A o autómato mínimo (a menos de um isomorfismo) para uma linguagem regular L. Mostre que a relação de Myhill-Nerode $\equiv_A$ associada a este autómato é idêntica à relação $\equiv_L$.

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

Resolução. As classes são $[\varepsilon]$, $[a]$ e $[b]$. Logo, L é regular.

O autómato mínimo é assim o seguinte:

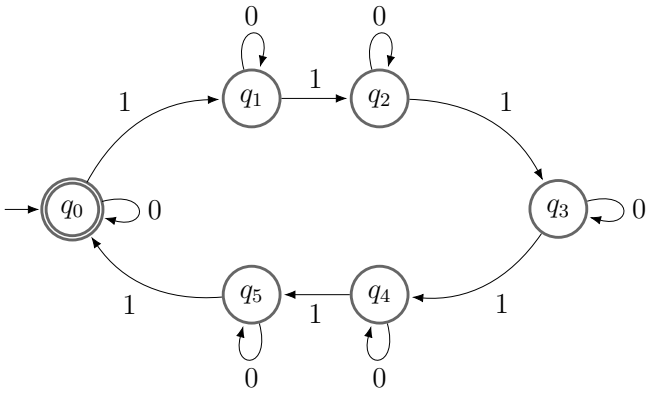


(b) $L = \{w \in \{0,1\}^\star : \ \text{a soma dos dígitos de } w \text{ é divisível por } 6\}$;

Resolução. Há 6 classes distintas:

- $[\varepsilon] = \{w \in \{0,1\}^\star : \ \#_1(w) \bmod 6 = 0\} = L$
- $[1] = \{w \in \{0,1\}^\star : \ \#_1(w) \bmod 6 = 1\}$
- $[11] = \{w \in \{0,1\}^\star : \ \#_1(w) \bmod 6 = 2\}$
- $[111] = \{w \in \{0,1\}^\star : \ \#_1(w) \bmod 6 = 3\}$
- $[1111] = \{w \in \{0,1\}^\star : \ \#_1(w) \bmod 6 = 4\}$
- $[11111] = \{w \in \{0,1\}^\star : \ \#_1(w) \bmod 6 = 5\}$

Sejam $q_0 = [\varepsilon]$, $q_1 = [1]$, $q_2 = [11]$, $q_3 = [111]$, $q_4 = [1111]$ e $q_5 = [11111]$. O autómato mínimo é o seguinte:



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

Resolução. Existe um número infinito de classes de equivalência. Se $i \neq j$ então $[a^i b] \neq [a^j b]$ (basta considerar $z = a^i$). Logo, L não é regular.

Não sendo regular, não existe qualquer autómato finito reconhecedor desta linguagem.