1 Autómatos finitos deterministas

1.1 Autómato finito determinista

Um alfabeto é um conjunto finito. Uma linguagem sobre um alfabeto I é um subconjunto de I^*. Usa-se simplesmente a designação linguagem sempre que não exista ambiguidade sobre qual é alfabeto em causa ou não seja relevante saber qual é o alfabeto.

Definição 1.1 AUTÓMATO FINITO DETERMINISTA
Um *autómato finito determinista*, ou apenas AFD, é um quíntuplo

$$D = (Q, I, \delta, q_0, F)$$

onde

- Q é um conjunto finito (*conjunto dos estados*);
- I é um conjunto finito (*conjunto dos símbolos de entrada*);
- $\delta : Q \times I \to Q$ é uma função (*função de transição*);
- $q_0 \in Q$ (*estado inicial*);
- $F \subseteq Q$ (*conjunto dos estados finais*). ◀

Qualquer AFD tem pelo menos um estado, o estado inicial q_0. O conjunto dos símbolos de entrada de um AFD é o alfabeto do AFD. Uma sequência de símbolos do alfabeto é neste contexto designada por palavra sobre o alfabeto ou, simplesmente, palavra. A sequência vazia, ϵ, é designada por palavra vazia. Note-se que a função de transição não tem de ser necessariamente uma função total. Um AFD pode ter um qualquer número de estados finais, em particular, nenhum.

Apresentam-se seguidamente vários exemplos de AFDS.

Exemplo 1.2 $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{p, q, r\}$;
- $I = \{a, b\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	a	b
p	q	
q	q	r
r	q	r

- $q_0 = p$;
- $F = \{q\}$.

Neste caso a função de transição δ não está definida para o par (p, b). ▲

Exemplo 1.3 $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{q_0, q_1\}$;
- $I = \{0, 1\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	0	1
q_0	q_0	q_1
q_1	q_1	q_0

- $F = \{q_0\}$. ▲

Exemplo 1.4 $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{\mathrm{P_0P_1}, \mathrm{P_0I_1}, \mathrm{I_0P_1}, \mathrm{I_0I_1}\}$;
- $I = \{0, 1\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	0	1
$\mathrm{P_0P_1}$	$\mathrm{I_0P_1}$	$\mathrm{P_0I_1}$
$\mathrm{P_0I_1}$	$\mathrm{I_0I_1}$	$\mathrm{P_0P_1}$
$\mathrm{I_0I_1}$	$\mathrm{P_0I_1}$	$\mathrm{I_0P_1}$
$\mathrm{I_0P_1}$	$\mathrm{P_0P_1}$	$\mathrm{I_0I_1}$

- $q_0 = \mathrm{P_0P_1}$;
- $F = \{\mathrm{I_0P_1}\}$. ▲

Exemplo 1.5 $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{q_0, q_1, q_2, q_3, q_4\}$;
- $I = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, .\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	0	1	2	3	4	5	6	7	8	9	.
q_0	q_1	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	
q_1											q_3
q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_3
q_3	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	
q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	

- $F = \{q_1, q_2, q_4\}$. ▲

Usa-se notação $\delta(q, a)\downarrow$ para indicar que a função de transição δ de um AFD atribui um valor a (q, a). Quando $\delta(q, a)\downarrow$ diz-se que existe uma transição a partir de q associada ao símbolo a e, se $\delta(q, a) = p$, diz-se que existe uma transição de q para p associada ao símbolo a. Usa-se a notação $\delta(q, a)\uparrow$ para

indicar que δ não está definida para o par (q, a). Quando $\delta(q, a)\uparrow$ diz-se que a partir de p não existe transição associada ao símbolo a.

Um caminho num AFD é uma sequência de estados $q_1 q_2 \dots q_n$, com $n > 0$, tal que, para cada $1 \leq k < n$, $\delta(q_k, a_k) = q_{k+1}$ para algum símbolo a_k do alfabeto, isto é, existe uma transição de q_k para q_{k+1}. As palavras $a_1 a_2 \dots a_{n-1}$ que assim se obtêm são as palavras que correspondem, ou que estão associadas, ao caminho $q_1 q_2 \dots q_n$. Por exemplo, considerando o AFD D apresentado no Exemplo 1.2, $pqrq$ e qrq são caminhos de D, sendo aba e ba palavras que lhes estão associadas, respectivamente. A um caminho num AFD podem corresponder várias palavras porque a partir de um estado podem existir transições associadas a diferentes símbolos para um mesmo estado. Como exemplo, considere-se o caminho $q_0 q_1 q_3 q_4$ do AFD apresentado no Exemplo 1.5. As palavras 0.0, 0.1 e 0.2 são exemplos de palavras associadas a este caminho. Observe-se que a partir do estado q_3 existem transições associadas a cada um dos dígitos de 0 a 9 para o estado q_4.

Por outro lado, dado um estado q e dada uma palavra $a_1 a_2 \dots a_n$, com $n \geq 0$, o caminho com início em q que corresponde, ou que está associado, a esta palavra é $q_1 q_2 \dots q_{n+1}$ em que $q_1 = q$ e $\delta(q_k, a_k) = q_{k+1}$, para cada $1 \leq k \leq n$. Note-se que este caminho pode não existir, mas, se existir, é único. Voltando ao AFD apresentado no Exemplo 1.2, o caminho com início em q que está associado à palavra baa é $qrqq$. Não está associado à palavra baa nenhum caminho com início em p.

Cada AFD define uma linguagem sobre o seu alfabeto. Para caracterizar rigorosamente esta linguagem é útil considerar a extensão a I^* da função de transição do AFD.

Considere-se fixado um AFD $D = (Q, I, \delta, q_0, F)$.

Definição 1.6 FUNÇÃO DE TRANSIÇÃO ESTENDIDA DE AFD
A *função de transição estendida* de D é a função $\delta^* : Q \times I^* \to Q$ tal que

$$\delta^*(q, w) = \begin{cases} q & \text{se } w = \epsilon \\ \delta^*(\delta(q, a), w') & \text{se } w = a.w' \text{ e } \delta(q, a)\downarrow \\ \text{não def} & \text{se } w = a.w' \text{ e } \delta(q, a)\uparrow \end{cases}$$

para cada $q \in Q$ e $w \in I^*$. ◀

Se $\delta^*(q, w) = q'$ tal significa que existe em D um caminho associado à palavra w que tem início em q e que este caminho termina em q'. Se w é $a_1 a_2 \dots a_n$, $n \geq 0$, o caminho é precisamente $q_1 q_2 \dots q_{n+1}$ em que $q_1 = q$, $q_{n+1} = q'$ e $q_{k+1} = \delta^*(q_1, a_1 \dots a_k)$, para cada $1 \leq k \leq n$.

Observe-se que se $\delta^*(q, w) = q'$ e $\delta^*(q', w') = q''$ então $\delta^*(q, w.w') = q''$ (Exercício **??** da secção **??**).

Definição 1.7 PALAVRA ACEITE E LINGUAGEM RECONHECIDA POR AFD
A palavra $w \in I^*$ diz-se *aceite* por D se $\delta^*(q_0, w) \in F$. O conjunto

$$L_D = \{w \in I^* : \delta^*(q_0, w) \in F\}$$

é a *linguagem reconhecida por* D, ou *linguagem de* D. ◀

A linguagem reconhecida por D é o conjunto das palavras aceites por D. As palavras aceites por D são precisamente as palavras associadas aos caminhos do autómato que começam com o estado inicial, q_0, e que terminam com um estado final.

Exemplo 1.8 Considere-se o autómato D apresentado no Exemplo 1.2. Calcule-se o valor que a função δ^* atribui ao par (p, aba):

$$\begin{aligned}\delta^*(p, aba) &= \delta^*(\delta(p, a), ba)\\ &= \delta^*(q, ba)\\ &= \delta^*(\delta(q, b), a)\\ &= \delta^*(r, a)\\ &= \delta^*(\delta(r, a), \epsilon)\\ &= \delta^*(q, \epsilon)\\ &= q\end{aligned}$$

O facto de $\delta^*(p, aba) = q$ significa que em D existe um caminho associado à palavra aba que tem início em p e que este caminho termina em q. O caminho é $pqrq$. Observe-se que os estados deste caminho são precisamente os estados que se vão sucessivamente obtendo ao calcular $\delta^*(p, aba)$. Como $\delta^*(p, aba) = q$ e q é um estado final, a palavra aba é uma palavra aceite por D e portanto aba pertence a L_D, a linguagem reconhecida por D.

Calcule-se agora o valor que a função δ^* atribui ao par $(p, aabb)$:

$$\begin{aligned}\delta^*(p, aabb) &= \delta^*(\delta(p, a), abb)\\ &= \delta^*(q, abb)\\ &= \delta^*(\delta(q, a), bb)\\ &= \delta^*(q, bb)\\ &= \delta^*(\delta(q, b), b)\\ &= \delta^*(r, b)\\ &= \delta^*(\delta(r, b), \epsilon)\\ &= \delta^*(r, \epsilon)\\ &= r\end{aligned}$$

O facto de $\delta^*(p, aabb) = r$ significa que em D existe um caminho associado à palavra $aabb$ que tem início em p e que este caminho termina em r. O caminho é $pqqrr$. Como $\delta^*(p, aabb) = r$ e r não é um estado final, a palavra $aabb$ não é uma palavra aceite por D e portanto a palavra $aabb$ não pertence a L_D.

Calcule-se ainda o valor que a função δ^* atribui ao par (p, baa): como $\delta(p, b)\uparrow$, isto é, não existe transição associada ao símbolo b a partir de p, tem-se que $\delta^*(p, baa)\uparrow$. Assim, à palavra baa não está associado nenhum caminho de D que comece no estado p. Uma vez que $\delta^*(p, baa)\uparrow$, $\delta^*(p, baa)$ não é, naturalmente, um estado final e portanto a palavra baa não é assim aceite por D. Consequentemente, baa não pertence a L_D. Observe-se que o facto de $\delta(p, b)\uparrow$ implica, em particular, que não sejam aceites por D palavras que comecem por b.

A linguagem reconhecida por D é o conjunto das palavras sobre o alfabeto $\{a, b\}$ que começam e terminam em a, dado que $\delta^*(p, w) \in F$ (ou seja, neste

caso, $\delta^*(p, w) = q$) se e apenas se w é uma palavra sobre $\{a, b\}$ que começa em a e termina em a. Com efeito, todos os caminhos de D que começam pelo estado inicial, o estado p, e terminam num estado final, neste caso o estado q, estão associados a palavras do alfabeto de D que começam e terminam em a e, vice-versa, a todas as palavras com esta propriedade estão associados caminhos que começam em p e terminam em q. ▲

Exemplo 1.9 Considere-se o autómato D apresentado no Exemplo 1.3. Observe-se que neste caso o estado inicial é também estado final. Calcule-se o valor que a função δ^* atribui ao par (q_0, ϵ):

$$\delta^*(q_0, \epsilon) \quad = q_0$$

Como q_0 é estado final, a palavra vazia é aceite por D. A palavra vazia pertence assim a L_D.

Calcule-se agora o valor que a função δ^* atribui ao par $(q_0, 1001)$

$$\begin{aligned}
\delta^*(q_0, 1001) &= \delta^*(\delta(q_0, 1), 001) \\
&= \delta^*(q_1, 001) \\
&= \delta^*(\delta(q_1, 0), 01) \\
&= \delta^*(q_1, 01) \\
&= \delta^*(\delta(q_1, 0), 1) \\
&= \delta^*(q_1, 1) \\
&= \delta^*(\delta(q_1, 1), \epsilon) \\
&= \delta^*(q_0, \epsilon) \\
&= q_0
\end{aligned}$$

Dado que $\delta^*(q_0, 1001) = q_0$ e q_0 é um estado final, a palavra 1001 é uma palavra aceite por D. A palavra 1001 pertence a L_D.

Calcule-se ainda o valor que a função δ^* atribui ao par $(q_0, 010)$

$$\begin{aligned}
\delta^*(q_0, 010) &= \delta^*(\delta(q_0, 0), 10) \\
&= \delta^*(q_0, 10) \\
&= \delta^*(\delta(q_0, 1), 0) \\
&= \delta^*(q_1, 0) \\
&= \delta^*(\delta(q_1, 0), \epsilon) \\
&= \delta^*(q_1, \epsilon) \\
&= q_1
\end{aligned}$$

Dado que $\delta^*(q_0, 010) = q_1$ e q_1 não é um estado final, a palavra 010 não é uma palavra aceite por D. A palavra 010 não pertence a L_D.

A linguagem reconhecida por D é o conjunto das palavras sobre o alfabeto $\{0, 1\}$ que têm um número par de 1's, pois, $\delta^*(q_0, w) \in F$ (ou seja, neste caso, $\delta^*(q_0, w) = q_0$) se e apenas se w é uma palavra do alfabeto de D que tem um número par de 1's. Observe-se que todos os caminhos de D que começam por q_0 e terminam no estado final, também q_0, estão associados a palavras do alfabeto de D que têm um número par de 1's e, vice-versa, todas as palavras com esta

propriedade estão associadas a caminhos de D que começam em q_0 e terminam em q_0. Note-se que a palavra vazia tem zero 1's, logo um número par de 1's ▲

Exemplo 1.10 A linguagem reconhecida pelo autómato D apresentado no Exemplo 1.4 é o conjunto das palavras sobre o alfabeto $\{0, 1\}$ que têm um número ímpar de 0's e um número par de 1's.

Neste autómato foram escolhidos para os estados nomes sugestivos com o propósito de facilitar a sua construção e a tarefa de perceber a linguagem que o autómato reconhece. A designação $\mathrm{P_0P_1}$ significa "número *par* de 0's e número *par* de 1's", a designação $\mathrm{P_0I_1}$ significa "número *par* de 0's e número *ímpar* de 1's", e os outros dois casos são semelhantes. Naturalmente estas são as quatro situações possíveis relativamente à paridade do número de 0's e 1's presentes numa sequência. A ideia subjacente à construção do autómato é que se um caminho começa no estado inicial e termina, por exemplo, no estado $\mathrm{P_0I_1}$ então todas as sequências a ele associadas têm um número *par* de 0's e número *ímpar* de 1's. Do mesmo modo, caso termine em $\mathrm{P_0P_1}$, as sequências a ele associadas têm de ter um número *par* de 0's e número *par* de 1's. Os outros casos são semelhantes. Assim, compreende-se que $\mathrm{I_0P_1}$ seja o (único) estado final. A escolha de $\mathrm{P_0P_1}$ para estado inicial corresponde ao facto de a sequência vazia ter zero 0's e zero 1's, logo um um número par de 0's e um número par de 1's. Compreende-se também que, por exemplo, $\delta(\mathrm{I_0P_1}, 0) = \mathrm{P_0P_1}$ (foi lido mais um 0 e portanto a paridade do número de 0's muda enquanto a de 1's se mantém) e que $\delta(\mathrm{I_0I_1}, 1) = \mathrm{I_0P_1}$ (muda agora a paridade do número de 1's).

Claro que do ponto de vista da linguagem reconhecida pelo autómato, o nome que se dá aos estados é irrelevante. Obtém-se um autómato que reconhece exactamente a mesma linguagem substituindo $\mathrm{P_0P_1}$, $\mathrm{P_0I_1}$, $\mathrm{I_0P_1}$ e $\mathrm{I_0I_1}$ por q_0, q_1, q_2 e q_3, respectivamente. ▲

Exemplo 1.11 A linguagem reconhecida pelo o autómato D apresentado no Exemplo 1.5 é o conjunto das constantes numéricas que têm uma parte inteira e uma parte decimal, separadas por ".", em que a parte inteira só começa por 0 se for exactamente 0 e a parte decimal pode ser vazia, caso em que "."é omitido. Mais rigorosamente, é o conjunto das constantes numéricas do tipo w_1 ou $w_1.w_2$ em que w_1, w_2 são sequências finitas e não vazias de dígitos de 0 a 9 e w_1 não começa por 0 excepto se $w_1 = 0$.

Assim, por exemplo, as palavras 35, 0.56 e 218.05 são aceites por D, mas as palavras 023 ou 35. não são aceites por D (verifique calculando os valores de δ^* apropriados). ▲

Note-se que se o estado inicial do AFD D for também estado final, a linguagem reconhecida por D inclui pelo menos uma palavra, a palavra vazia ϵ. Por outro lado, para que ϵ seja aceite pelo AFD, o estado inicial tem necessariamente que ser estado final.

Proposição 1.12 *O* AFD D *aceita a palavra vazia,* ϵ*, se e só se* $q_0 \in F$*.*

Prova: Tem-se que $\delta^*(q_0, \epsilon) = q_0$ e ϵ é aceite por D se e só se $\delta^*(q_0, \epsilon) \in F$. ∎

Observe-se ainda que se o conjunto dos estados finais do AFD D é o conjunto vazio, nenhuma palavra é aceite por D. A linguagem reconhecida pelo AFD é neste caso a linguagem vazia, isto é, $L_D = \emptyset$.

Do ponto de vista da linguagem reconhecida por um AFD, o nome que se dá aos estados é irrelevante. Assim, dados dois AFDS, pode sempre assumir-se, sem perda de generalidade, que os respectivos conjuntos de estados são disjuntos.

Dada um certa linguagem L, diz-se que D é um AFD para L se D é um AFD e $L_D = L$. Diz-se que uma certa linguagem L é reconhecida por um AFD sempre que exista um AFD para L. As linguagens que são reconhecidas por AFDS têm uma designação especial: são as linguagens regulares.

Definição 1.13 LINGUAGEM REGULAR
Uma linguagem L diz-se *regular* se existe um AFD D tal que $L_D = L$. ◀

Exemplo 1.14 Tendo em conta os exemplos anteriores são linguagens regulares, por exemplo, as seguintes:

- o conjunto das palavras sobre o alfabeto $\{a, b\}$ que começam e terminam em a;
- o conjunto das palavras sobre o alfabeto $\{0, 1\}$ que têm um número ímpar de 1's;
- o conjunto das palavras sobre o alfabeto $\{0, 1\}$ que têm um número par de 0's e um número ímpar de 1's;
- o conjunto das constantes numéricas do tipo w_1 ou $w_1.w_2$ em que w_1, w_2 são sequências finitas e não vazias de dígitos de 0 a 9 e w_1 não começa por 0 excepto se $w_1 = 0$.

Exemplos de linguagens não regulares são as seguintes:

- o conjunto das palavras sobre o alfabeto $\{a, b\}$ nas quais o número de a's é o dobro do número de b's;
- expressões aritméticas nas quais o número de parêntesis esquerdos é igual ao número de parêntesis direitos.

A prova de que estas linguagens não são de facto regulares será apresentada adiante. ▲

A terminar esta secção, merece referência a seguinte observação a propósito da definição de AFD adoptada neste texto. Na definição adoptada, a função de transição não tem de ser necessariamente uma função total. Veja-se, por exemplo, os AFDS apresentados nos Exemplos 1.2 e 1.5. Mas outros autores incluem na definição de AFD a exigência de que a função de transição tem de ser total. No entanto, as duas definições são equivalentes, no que diz respeito às linguagens que são reconhecidas pelos dois tipos de AFDS. Qualquer linguagem que seja reconhecida por um AFD com função de transição não total é também reconhecida por um AFD com função de transição total.

Dado um AFD D com função de transição não total, obtém-se facilmente um novo AFD com função de transição total cuja linguagem é L_D. É um AFD igual a D no que respeita ao alfabeto, estado inicial e estados finais, mas tem mais um estado, o estado q_t. Sempre que em D não exista uma transição associada a um símbolo a a partir de um estado q, no novo autómato vai existir uma transição associada a a de q para q_t. A função de transição deste novo autómato tem de ser total, logo têm de existir transições associadas a cada símbolo do alfabeto a partir de q_t. Como se pretende que a linguagem reconhecida não seja alterada, estas transições são transições para o próprio q_t. Um AFD construído deste modo é designado por totalização de D. Dado um AFD D com função de transição total, a totalização de D é, naturalmente, o próprio autómato D.

Definição 1.15 TOTALIZAÇÃO DE AFD
Uma *totalização* de D é um AFD $TOT(D)$ construído como se segue:

- se a função de transição de D é total, $TOT(D) = D$;
- se a função de transição de D não é total, $TOT(D) = (Q_\mathrm{t}, I, \delta_\mathrm{t}, q_0^\mathrm{t}, F_\mathrm{t})$ em que
  - $Q_\mathrm{t} = Q \cup \{q_\mathrm{t}\}$ com $q_\mathrm{t} \notin Q$;
  - $\delta_\mathrm{t} : Q_\mathrm{t} \times I \to Q_\mathrm{t}$ é tal que

$$\delta_\mathrm{t}(q,a) = \begin{cases} \delta(q,a) & \text{se } q \in Q \text{ e } \delta(q,a)\downarrow \\ \\ q_\mathrm{t} & \text{se } q = q_\mathrm{t} \text{ ou } \delta(q,a)\uparrow \end{cases}$$

    para cada $q \in Q_\mathrm{t}$ e $a \in I$;
  - $q_0^\mathrm{t} = q_0$;
  - $F_\mathrm{t} = F$. ◀

Segue-se dois exemplos que ilustram a construção de totalizações de AFDS.

Exemplo 1.16 Seja D o AFD apresentado no Exemplo 1.3. Dado que a sua função de transição é total, $TOT(D) = D$. ▲

Exemplo 1.17 Seja D o AFD apresentado no Exemplo 1.5. Dado que a sua função de transição não é total, $TOT(D) = (Q_\mathrm{t}, I, \delta_\mathrm{t}, q_0^\mathrm{t}, F_\mathrm{t})$ em que

- $Q_\mathrm{t} = \{q_0, q_1, q_2, q_3, q_4, q_\mathrm{t}\}$;
- $I = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, .\}$;
- $\delta_\mathrm{t} : Q \times I \to Q_\mathrm{t}$ é tal que

δ_t	0	1	2	3	4	5	6	7	8	9	.
q_0	q_1	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_t
q_1	q_t	q_t	q_t	q_t	q_t	q_t	q_t	q_t	q_t	q_t	q_3
q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_2	q_3
q_3	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_t
q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_4	q_t
q_t	q_t	q_t	q_t	q_t	q_t	q_t	q_t	q_t	q_t	q_t	q_t

- $q_0^{\mathrm{t}} = q_0$;
- $F_{\mathrm{t}} = F = \{q_1, q_2, q_4\}$. ▲

Apresenta-se agora a proposição que estabelece que a linguagem de $TOT(D)$ é de facto L_D.

Proposição 1.18 *As linguagens de D e $TOT(D)$ são iguais.*

Prova: Esta prova é deixada como exercício ao leitor. ■

1.2 Autómatos vacuosos e triviais

Como já foi referido, se o conjunto dos estados finais de um AFD D é o conjunto vazio então a linguagem reconhecida por D é a linguagem vazia, isto é, $L_D = \emptyset$. Mas também existem AFDS com estados finais cuja linguagem é vazia: basta que não exista nenhum caminho que comece no estado inicial e termine num estado final. É o caso do AFD apresentado no exemplo seguinte.

Exemplo 1.19 $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{p, q, r, s\}$;
- $I = \{a, b\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	a	b
p	q	
q	r	q
r		r
s	q	r

- $q_0 = p$;
- $F = \{s\}$.

Note-se que não existe neste autómato nenhum caminho que comece com p (estado inicial) e termine com s (o único estado final). Assim, $L_D 0 \emptyset$. ▲

Designa-se por AFD vacuoso qualquer AFD cuja linguagem seja vazia. Os AFDS mais simples com alfabeto I cuja linguagem é vazia são os que têm um único estado, o estado inicial, a sua função de transição não está definida para nenhum elemento do seu conjunto de partida e não têm estados finais . Estes AFDS são AFDS vacuosos canónicos. Obviamente todos eles são iguais, a menos do nome dado ao seu estado.

Definição 1.20 AFD VACUOSO E AFD VACUOSO CANÓNICO
O AFD D é *vacuoso* se a sua linguagem é a linguagem vazia. Um AFD *vacuoso canónico* é $D_{\mathrm{vac}} = (\{q\}, I, \delta_{\mathrm{vac}}, q, \emptyset)$ em que $\delta_{\mathrm{vac}}(q, a)\uparrow$ para cada $a \in I$. ◀

A linguagem reconhecida por um AFD pode também ser o conjunto de todas as palavras sobre o seu alfabeto I. É o caso do autómato apresentado no exemplo seguinte.

Exemplo 1.21 $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{p, q, r\}$;
- $I = \{a, b\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	a	b
p	r	q
q	p	q
r	r	p

- $q_0 = p$;
- $F = \{p, q, r\}$.

Observe-se que neste autómato a qualquer sequência em $\{a, b\}^*$ está associado um caminho que começa no estado inicial, p, e termina num estado final. A linguagem de D é assim $\{a, b\}^*$. ▲

Designa-se por AFD trivial qualquer AFD com esta propriedade. Os AFDS mais simples com alfabeto I cuja linguagem é I^* são os que têm um único estado, que é estado inicial e final, e têm uma função de transição que é função total. Estes AFDS são designados AFDS triviais canónicos.

Definição 1.22 AFD TRIVIAL
O AFD D é *trivial* se a sua linguagem é o conjunto I^*. Um AFD *trivial canónico* é $D_{\text{triv}} = (\{q\}, I, \delta_{\text{triv}}, q, \{q\})$ em que $\delta_{\text{triv}}(q, a) = q$ para cada $a \in I$. ◀

A proposição seguinte estabelece que a linguagem dos AFDS vacuosos canónicos é a linguagem vazia e que a linguagem dos AFDS triviais canónicos é o conjunto de todas as sequências sobre o seu alfabeto.

Proposição 1.23

1. *A linguagem de um* AFD *vacuoso canónico é a linguagem vazia.*
2. *A linguagem de um* AFD *trivial canónico com alfabeto I é I^*.* ■

1.3 Estados acessíveis, produtivos, úteis e inúteis

Num AFD existem certos estados notáveis que verificam propriedades relevantes para o estudo desse AFD e da linguagem por ele reconhecida. Nesta secção faz-se referência a quatro tipos de estados notáveis: estados acessíveis, estados produtivos, estados úteis e estados inúteis.

Definição 1.24 Estado acessível, produtivo, útil e inútil de afd
Diz-se que um estado q de D é

- *acessível* se existe $w \in I^*$ tal que $\delta^*(q_0, w) = q$;
- *produtivo* se existe $w \in I^*$ tal que $\delta^*(q, w) \in F$;
- *útil* se é estado acessível e é estado produtivo;
- *inútil* se não é estado útil.

Denota-se por Ac_D o conjunto dos estados acessíveis de D, por Prd_D o conjunto dos estados produtivos de D, por Ut_D o conjunto dos estados úteis de D e por In_D o conjunto dos estados inúteis de D. ◀

Um estado q do autómato D é acessível se existe um caminho de D que começa no estado inicial e termina em q. O estado q é produtivo se existe um caminho de D que tem início em q e termina num estado final. O estado q é inútil se não for acessível ou não for produtivo. Obviamente, $Ut_D = Ac_D \cap Prd_D$ e $In_D = Q \setminus Ut_D$.

Exemplo 1.25 Considere-se o afd $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{p, q, r, s\}$;
- $I = \{a, b\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	a	b
p	q	s
q	q	r
r	q	r
s	s	s

- $q_0 = p$;
- $F = \{q\}$.

Os estados p, q e r são acessíveis e produtivos. São deste modo estados úteis. O estado s é acessível mas não é produtivo, logo é um estado inútil. Deste modo, $Ac_D = Q$, $Prd_D = Ut_D = \{p, q, r\}$ e $In_D = \{s\}$.

Note-se que a linguagem reconhecida por este autómato é o conjunto de todas as sequências de a's e b's que começam e terminam em a. É exactamente a mesma linguagem que é reconhecida pelo autómato apresentado no Exemplo 1.2. O estado s incluído no autómato acima não contribui para alterar o conjunto das sequências aceites. Com efeito, não existem neste autómato caminhos que incluam simultaneamente s e um estado final. Daqui decorre a designação de inútil que aqui é atribuída a s. ▲

Exemplo 1.26 Considere-se o afd $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{q_0, q_1, q_2\}$;
- $I = \{0, 1\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	0	1
q_0	q_0	q_1
q_1	q_1	q_0
q_2	q_0	q_1

- $F = \{q_0\}$.

Os estados q_0 e q_1 são acessíveis e produtivos. São deste modo estados úteis. O estado q_2 é produtivo mas não é acessível, logo é um estado inútil. Deste modo, $Ac_D = Ut_D = \{q_0, q_1\}$, $Prd_D = Q$ e $In_D = \{q_2\}$.

Note-se que a linguagem reconhecida por este autómato é o conjunto de todas as sequências de 0's e 1's que têm um número par de 1's. É exactamente a mesma linguagem que é reconhecida pelo autómato apresentado no Exemplo 1.3. O estado q_2 incluído no autómato acima não contribui para alterar o conjunto de sequências aceites. Com efeito, não existem neste autómato caminhos que incluam simultaneamente o estado inicial e q_2. Daqui decorre a designação de inútil aqui atribuída a q_2. ▲

Exemplo 1.27 Todos os estados dos autómatos apresentados nos Exemplos 1.2, 1.3, 1.4 e 1.5 são acessíveis e produtivos. Deste modo, todos esses estados são estados úteis. ▲

O estado inicial de D é sempre um estado acessível, uma vez que $\delta^*(q_0, \epsilon) = q_0$. Assim, o estado inicial só é inútil se não for produtivo. Quando isto acontece, não existe nenhuma palavra $w \in I^*$ tal que $\delta^*(q_0, w)$ seja um estado final e portanto a linguagem reconhecida por D é a linguagem vazia, ou seja, D é vacuoso. Por outro lado, os estados finais são sempre estados produtivos. Assim, um estado final só é inútil se não for acessível. Se todos os estados finais de D são inúteis então D é também vacuoso.

Proposição 1.28 *São equivalentes as seguintes afirmações sobre o* AFD D*:*

(i) D *é vacuoso;*

(ii) o estado inicial de D *não é produtivo;*

(iii) nenhum estado final de D *é acessível;*

(iv) todos os estados de D *são inúteis.* ■

Como se verá adiante com mais detalhe, os estados inúteis de um autómato podem, em geral, ser removidos, obtendo-se um autómato que reconhece exactamente a mesma linguagem. Como exemplo, refiram-se os autómatos apresentados nos Exemplos 1.25 e 1.27. O único caso em que nem todos os estados inúteis podem ser removidos é o caso em que D é um AFD vacuoso. Neste caso,

todos os estados de D são inúteis. Não é possível remover todos estes estados porque um AFD tem de ter pelo menos um estado, o estado inicial. Assim, todos os estados são removidos com a excepção do estado inicial, obtendo-se um AFD vacuoso canónico.

O caso dos estados acessíveis mas não produtivos merece uma nota complementar. Estes estados podem, em geral, ser removidos porque a função de transição δ não tem necessariamente que ser uma função total.

Apresenta-se seguidamente um algoritmo que permite calcular os estados notáveis acima referidos, isto é, os estados acessíveis, os estados produtivos e os estados úteis e inúteis de um AFD.

Definição 1.29 ALGORITMO DE PROCURA DE ESTADOS NOTÁVEIS (APEN) O algoritmo de procura de estados notáveis (APEN) é o seguinte:

ENTRADA: AFD $D = (Q, I, \delta, q_0, F)$
SAÍDA: o tuplo de conjuntos $APEN(D) = (Ac, Prd, Ut, In)$

1. $A := \{q_0\}$;
2. $Aux := \bigcup_{a \in I} \{\delta(q_0, a)\}$;
3. enquanto $Aux \not\subseteq A$
   3.1. $A := A \cup Aux$;
   3.2. $Aux := \bigcup_{a \in I} \{\delta(p, a) : p \in Aux\}$;
4. $Ac := A$;
5. $A := F$;
6. $Aux := \bigcup_{a \in I} \{p : \delta(p, a) \in F\}\}$;
7. enquanto $Aux \not\subseteq A$
   7.1. $A := A \cup Aux$;
   7.2. $Aux := \bigcup_{a \in I} \{p : \delta(p, a) \in Aux\}$;
8. $Prd := A$;
9. $Ut := Ac \cap Prd$;
10. $In := Q \backslash (Ac \cap Prd)$. ▲

A execução do APEN termina sempre e calcula o conjunto de todos os estados acessíveis, o conjunto de todos os estados produtivos, o conjunto de todos os estados úteis e o conjunto de todos os estados inúteis do AFD de entrada. Esta propriedade do APEN é enunciada na Proposição 1.30.

Proposição 1.30 *A execução do APEN termina sempre e, sendo D o* AFD *de entrada e $APEN(D) = (Ac, Prd, Ut, In)$ a informação de saída, tem-se que $Ac = Ac_D$, $Prd = Prd_D$, $Ut = Ut_D$ e $In = In_D$.* ■

1.4 Produto e complementação

Dados dois AFDS D_1 e D_2 como o mesmo alfabeto, pode construir-se um novo AFD, o AFD produto $D_1 \times D_2$, que tem a particularidade de reconhecer exactamente a linguagem constituídas pelas palavras que são simultaneamente aceites por D_1 e por D_2, ou seja, a linguagem $L_{D_1} \cap L_{D_2}$.

Uma razão pela qual esta construção é interessante é porque permite a construção modular de AFDS. Por construção modular entende-se o seguinte:

se o objectivo é obter um AFD para uma linguagem que tem de satisfazer, por exemplo, dois requisitos em simultâneo, começa-se por construir um AFD para a linguagem que satisfaz apenas um dos requisitos e um AFD para a linguagem que satisfaz apenas o outro (que serão, em geral, mais simples que o AFD final pretendido), e constrói-se, por fim, o seu autómato produto.

Uma outra razão pela qual esta construção é relevante, é o facto de permitir concluir que as linguagens regulares têm uma propriedade interessante: a linguagem que resulta da intersecção de duas linguagens regulares é também regular. Por esta razão diz-se que o conjunto das linguagens regulares é fechado para a intersecção.

Apresenta-se agora a definição de AFD produto, $D_1 \times D_2$, na qual se pressupõe que os AFDS D_1 e D_2 têm o mesmo alfabeto. A linguagem é a intersecção das linguagens dos autómatos dados.

Definição 1.31 AFD PRODUTO
Dados dois AFDS $D_1 = (Q_1, I, \delta_1, q_0^1, F_1)$ e $D_2 = (Q_2, I, \delta_2, q_0^2, F_2)$, o AFD *produto* de D_1 e D_2 é o AFD $D_1 \times D_2 = (Q, I, \delta, q_0, F)$ em que

- $Q = Q_1 \times Q_2$;
- $\delta : Q \times I \to Q$ é tal que

$$\delta((q_1, q_2), a) = \begin{cases} (\delta_1(q_1, a), \delta_2(q_2, a)) & \text{se } \delta_1(q_1, a)\downarrow \text{ e } \delta_2(q_2, a)\downarrow \\ \text{não def} & \text{caso contrário} \end{cases}$$

  para cada $(q_1, q_2) \in Q$ e $a \in I$;
- $q_0 = (q_0^1, q_0^2)$;
- $F = F_1 \times F_2$. ◀

O exemplo seguinte ilustra a construção do autómato produto.

Exemplo 1.32 Considerem-se os AFDS D_1 e D_2 com alfabeto $I = \{0, 1\}$

- $D_1 = (Q_1, I, \delta_1, q_0^1, F_1)$ em que
  - $Q_1 = \{r_0, r_1\}$;
  - $\delta_1 : Q_1 \times I \to Q_1$ é tal que

δ	0	1
r_0	r_0	r_1
r_1	r_1	r_0

  - $q_0^1 = r_0$;
  - $F = \{r_0\}$.
- $D_2 = (Q_2, I, \delta_2, q_0^2, F_2)$ em que
  - $Q_2 = \{s_0, s_1\}$;

– $\delta_2 : Q_2 \times I \to Q_2$ é tal que

δ	0	1
s_0	s_1	s_0
s_1	s_0	s_1

– $q_0^2 = s_0$;
– $F_2 = \{s_1\}$.

O AFD produto é $D_1 \times D_2 = (Q, I, \delta, q_0, F)$ em que

- $Q = \{(r_0, s_0), (r_0, s_1), (r_1, s_0), (r_1, s_1)\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	0	1
(r_0, s_0)	(r_0, s_1)	(r_1, s_0)
(r_0, s_1)	(r_0, s_0)	(r_1, s_1)
(r_1, s_0)	(r_1, s_1)	(r_0, s_0)
(r_1, s_1)	(r_1, s_0)	(r_0, s_1)

- $q_0 = (r_0, s_0)$;
- $F = \{(r_0, s_1)\}$.

Observe-se que o AFD D_1 é, a menos de uma modificação no nome dos estados, o AFD apresentado no Exemplo 1.3. A linguagem L_{D_1} é assim o conjunto das palavras sobre o alfabeto $\{0, 1\}$ que têm um número par de 1's. Por outro lado, é fácil perceber que a linguagem L_{D_2} é o conjunto das palavras sobre o mesmo alfabeto que têm um número ímpar de 0's.

Relativamente ao AFD produto $D_1 \times D_2$, comece-se por observar que, seguindo a definição de AFD produto, o estado inicial é o par constituído pelos estados iniciais dos dois autómatos e o (único) estado final é o par constituído pelo (único) estado final de D_1 e o (único) estado final de D_2. Observe-se também que no AFD produto existe uma transição associada a um símbolo do alfabeto de um estado q para um outro q' se, e apenas se, em D_1 existe uma transição associada a esse símbolo do estado de D_1 em q para o estado de D_1 em q', e o mesmo acontece em D_2.

Cada caminho de $D_1 \times D_2$ pode ser visto como tendo sido obtido a partir de um caminho de D_1 e de outro de D_2, e se uma palavra está associada a esse caminho de $D_1 \times D_2$ então ela está associada quer ao correspondente caminho de D_1, quer ao correspondente caminho de D_2. Considerando o caminho $(r_0, s_0)(r_0, s_1)(r_1, s_1)(r_0, s_1)$, por exemplo, o caminho de D_1 é $r_0 r_0 r_1 r_0$ e o de D_2 é $s_0 s_1 s_1 s_1$. Ao caminho do autómato produto está associada a palavra 011 e o mesmo acontece relativamente aos caminhos de D_1 e D_2.

Por outro lado, um caminho de D_1 e um caminho de D_2 só vão dar origem a um caminho de $D_1 \times D_2$ se existir uma palavra que esteja associada quer a D_1 quer a D_2, estando essa palavra também associada ao caminho do autómato produto. Considerem-se, por exemplo, $r_0 r_0 r_0 r_0$ de D_1 e $s_0 s_1 s_0 s_1$ de D_2. A

ambos corresponde a palavra 000. Existe no autómato produto um caminho obtido a partir destes, o caminho $(r_0, s_0)(r_0, s_1)(r_0, s_0)(r_0, s_1)$, ao qual está associada 000. Mas a $r_0r_0r_1r_0$ apenas corresponde 011 e a $s_0s_0s_1s_1$ apenas corresponde 101 e $(r_0, s_0)(r_0, s_0)(r_1, s_1)(r_0, s_1)$ não é caminho de $D_1 \times D_2$.

A linguagem $L_{D_1 \times D_2}$ é o conjunto das palavras sobre o alfabeto $\{0, 1\}$ que tem um número ímpar de 0's e um número par de 1's. Compare-se $D_1 \times D_2$ como o AFD apresentado no Exemplo 1.4. Estes dois autómatos são, de facto, iguais, a menos do nome dado aos estados. ▲

O Exemplo 1.32 anterior ilustra a utilização do AFD produto na construção modular de AFDS. Obteve-se um AFD para a linguagem constituída pelas palavras sobre $\{0, 1\}$ que satisfazem dois requisitos, terem um número ímpar de 0's e terem um número par de 1's, a partir de dois AFDS mais simples, um para a linguagem das palavras que têm um número ímpar de 0's e outro para a linguagem das palavras que têm um número par de 1's.

A Proposição seguinte estabelece que a linguagem do AFD $D_1 \times D_2$ é precisamente a intersecção das linguagens L_{D_1} e L_{D_2}.

Proposição 1.33 Sejam D_1 e D_2 AFDS com o mesmo alfabeto e $D_1 \times D_2$ o AFD produto. Tem-se que $L_{D_1 \times D_2} = L_{D_1} \cap L_{D_2}$. ■

O resultado anterior estabelece que sendo D_1 e D_2 AFDS com o *mesmo* alfabeto, a linguagem de $D_1 \times D_2$ é a intersecção das linguagens de D_1 e de D_2. No caso de $D_1 = (Q_1, I_1, \delta_1, q_0^1, F_1)$ e $D_2 = (Q_2, I_2, \delta_2, q_0^2, F_2)$ com $I_1 \neq I_2$, é fácil perceber que é também possível obter a partir de D_1 e D_2 um novo AFD cuja linguagem é $L_{D_1 \times D_2}$. Basta considerar $D_1' = (Q_1, I_1 \cup I_2, \delta_1', q_0^1, F_1)$ e $D_2' = (Q_2, I_1 \cup I_2, \delta_2', q_0^2, F_2)$ em que δ_1' é a extensão de δ_1 a $Q_1 \times (I_1 \cup I_2)$ que não está definida para os elementos não pertencentes a $Q_1 \times I_1$ e δ_2' é definida de modo semelhante. Naturalmente, $L_{D_1'} = L_{D_1}$ e $L_{D_2'} = L_{D_2}$ e, portanto, $L_{D_1' \times D_2'} = L_{D_1'} \cap L_{D_2'} = L_{D_1 \times D_2}$.

Dado um AFD D com alfabeto I, é também fácil construir a partir de D um outro AFD que aceita precisamente todas as palavras de I^* que não são aceites por D, ou seja, um AFD cuja linguagem seja $I^* \backslash L_D$, o conjunto complementar de L_D relativamente a I^*. Este novo autómato é o AFD complementar de D, o AFD $\overline{D}$.

Esta construção é também útil na construção modular de autómatos, como se ilustrará adiante. Uma outra razão pela qual esta construção é relevante é o facto de permitir concluir que as linguagens regulares têm uma outra propriedade interessante: dada uma qualquer linguagem regular L sobre um alfabeto I, a linguagem complementar de L relativamente a I^*, $I^* \backslash L$, é também uma linguagem regular. Isto significa que o conjunto das linguagens regulares tem uma outra propriedade de fecho: o conjunto das linguagens regulares é fechado para a complementação.

Quando a função de transição do AFD D é total, é fácil perceber que se obtém um AFD para a linguagem $I^* \backslash L_D$ considerando um autómato idêntico a D mas em que os estados finais são precisamente todos os estados não finais de D. Quando a função de transição de D não é total, bastará considerar primeiro

um autómato totalização de D, $TOT(D)$, que tem função de transição total e cuja linguagem é precisamente L_D, e depois proceder como no caso anterior.

Definição 1.34 AFD COMPLEMENTAR
Dado um AFD $D = (Q, I, \delta, q_0, F)$, um AFD *complementar* de D é um AFD $\overline{D}$ construído como se segue:

- $\overline{D} = (Q, I, \delta, q_0, Q \backslash F)$ se a função de transição δ é função total;
- $\overline{D} = (Q_\mathrm{t}, I, \delta_\mathrm{t}, q_0^\mathrm{t}, Q_\mathrm{t} \backslash F_\mathrm{t})$ se a função de transição δ não é função total e $TOT(D) = (Q_\mathrm{t}, I, \delta_\mathrm{t}, q_0^\mathrm{t}, F_\mathrm{t})$ é uma totalização de D. ◀

Apresentam-se seguidamente dois exemplos ilustrativos da construção de AFDS complementares.

Exemplo 1.35 Considere-se o AFD D apresentado no Exemplo 1.3. O AFD complementar é $\overline{D} = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_1\})$ com

δ	0	1
q_0	q_0	q_1
q_1	q_1	q_0

Como δ é função total, o AFD complementar $\overline{D}$ é obtido mudando apenas os estados finais: o conjunto dos estados finais de $\overline{D}$ é o conjunto dos estados não finais de D, isto é, $\{q_1\}$. Como todas as outras componentes se mantêm, os caminhos de D e de $\overline{D}$ são exactamente os mesmos e, em particular, as palavras associadas aos caminhos com início no estado inicial, q_0, são também exactamente as mesmas nos dois autómatos. Se um caminho com início em q_0 termina no estado final de D, também q_0, então a palavra correspondente é aceite por D. Como esse estado já não é estado final em $\overline{D}$, a palavra não é aceite por $\overline{D}$. Por exemplo, o caminho $q_0q_0q_1q_0$ está associado à palavra 011. Esta sequência é aceite por D, mas não é aceite por $\overline{D}$.

Se um caminho com início em q_0 não termina no estado final de D, isto é, termina em q_1, a palavra correspondente não é aceite por D e, como este estado é agora estado final de $\overline{D}$, a palavra é aceite por $\overline{D}$. Por exemplo, ao caminho $q_0q_0q_1q_0q_1$ está associada a palavra 0111. Esta sequência não é aceite por D, mas é aceite por $\overline{D}$.

A linguagem de $\overline{D}$ é assim o conjunto de todas as palavras sobre $\{0, 1\}^*$ que não são aceites por D, isto é, $L_{\overline{D}}$ é o complementar de L_D relativamente a $\{0, 1\}^*$. Consequentemente, $L_{\overline{D}}$ é o conjunto de todas as palavras sobre $\{0, 1\}^*$ que têm um número ímpar de 1's. ▲

Exemplo 1.36 Considere-se o AFD D apresentado no Exemplo 1.2. Dado que a função de transição não é total a construção de um AFD complementar, $\overline{D}$ tem dois passos:

(i) $TOT(D) = (Q_\mathrm{t}, I, \delta_\mathrm{t}, q_0^\mathrm{t}, F_\mathrm{t})$ em que

- $Q_\mathrm{t} = \{p, q, r, q_\mathrm{t}\}$;

– $I = \{a, b\}$;

– $\delta_{\mathrm{t}} : Q_{\mathrm{t}} \times I \to Q_{\mathrm{t}}$ é tal que

δ_{t}	a	b
p	q	q_{t}
q	q	r
r	q	r
q_{t}	q_{t}	q_{t}

– $q_0^{\mathrm{t}} = p$;

– $F_{\mathrm{t}} = F = \{q\}$.

(ii) $\overline{D} = (Q_{\mathrm{t}}, I, \delta_{\mathrm{t}}, q_0^{\mathrm{t}}, Q_{\mathrm{t}} \backslash F_{\mathrm{t}})$, isto é, $\overline{D} = (\{p, q, r, q_{\mathrm{t}}\}, \{a, b\}, \delta_{\mathrm{t}}, p, \{p, r, q_{\mathrm{t}}\})$.

O AFD complementar de D, $\overline{D}$, considerado é em tudo semelhante a $TOT(D)$ excepto no que respeita aos estados finais: o conjunto dos estados finais de $\overline{D}$ é $Q_{\mathrm{t}} \backslash F$. Assim os estados finais de $\overline{D}$ são precisamente os estados não finais de D e o novo estado q_{t} introduzido ao efectuar a totalização de D.

Em particular, note-se que para cada palavra $w \in \{a, b\}^*$ que não é aceite por D pelo facto de $\delta^*(p, w)\uparrow$, ou seja, para cada palavra começada por b, vai existir um caminho de $\overline{D}$ associado a w que começa em p e que termina em q_{t}. Como q_{t} é estado final, w é aceite por este autómato. Por exemplo, *baba* não é aceite por D, pois $\delta^*(p, baba)\uparrow$ mas, em $\overline{D}$, existe o caminho $pq_{\mathrm{t}}q_{\mathrm{t}}q_{\mathrm{t}}q_{\mathrm{t}}$ ao qual corresponde a palavra *baba*, que é assim aceite por $\overline{D}$. ▲

A proposição seguinte estabelece que a linguagem do AFD $\overline{D}$ é de facto a linguagem complementar de L_D relativamente ao conjunto das palavras sobre o alfabeto de D.

Proposição 1.37 Seja D um AFD e $\overline{D}$ um seu complementar. Tem-se que $L_{\overline{D}} = I^* \backslash L_D$.

Prova: Seja $D = (Q, I, \delta, q_0, F)$ o AFD dado e $\overline{D}$ como na Definição 1.34.

(1) No caso de δ ser função total, tem-se, para cada $w \in I^*$, $\delta^*(q_0, w) \in L_{\overline{D}}$ se e só se $\delta^*(q_0, w) \in Q \backslash F$ se e só se $\delta^*(q_0, w) \notin Q \backslash F$ se e só se $\delta^*(q_0, w) \in L_D$. Logo, $L_{\overline{D}} = I^* \backslash L_D$.

(2) Trata-se agora o caso em que δ não é função total. A função de transição de $TOT(D)$ é função total e, pela Proposição 1.18, $L_{TOT(D)} = L_D$. Fazendo um raciocínio idêntico ao apresentado em (1), mas usando $TOT(D)$ em vez de D, conclui-se que $L_{\overline{D}} = I^* \backslash L_D$. ■

A terminar esta secção, apresentam-se alguns exemplos que ilustram a utilização dos autómatos produto e complementação na construção modular de AFDS, no sentido acima referido.

Exemplo 1.38 Seja L a linguagem sobre o alfabeto $\{0, 1\}$ constituída pelas palavras que têm um número ímpar de 0's e um número par de 1's com a excepção das palavras que têm dois 0's consecutivos. Por exemplo, $01010 \in L$ e $011 \in L$, mas $1011 \notin L$ (porque a palavra tem um número ímpar de 1's) e

$00110 \notin L$ (porque, apesar da paridade do número 0's e 1's estar correcta, a palavra tem dois 0's consecutivos).

Pretende-se construir um AFD D para a linguagem L. Note-se que designando por L' o conjunto das palavras sobre $\{0,1\}$ que têm um número ímpar de 0's e um número par de 1's e por L'' o conjunto das palavras sobre o mesmo alfabeto que têm dois 0's consecutivos, tem-se que $L = L' \backslash L''$, ou, de modo equivalente, $L = L' \cap (\{0,1\}^* \backslash L'')$. Assim, usando o produto e a complementação de AFDS, pode obter-se um AFD para L da seguinte forma: (i) constrói-se um AFD D' para L'; (ii) constrói-se um AFD D'' para L''; (iii) constrói-se $\overline{D''}$; (iv) constrói-se $D' \times \overline{D''}$. Recorde-se que D' pode, por sua vez, ser obtido através do produto de AFDS (Exemplo 1.32).

(i) $D' = (Q', \{0,1\}, \delta', q_0', F')$ em que

- $Q' = \{(r_0, s_0), (r_0, s_1), (r_1, s_0), (r_1, s_1)\}$;
- $\delta' : Q' \times \{0,1\} \to Q'$ é tal que

δ'	0	1
(r_0, s_0)	(r_0, s_1)	(r_1, s_0)
(r_0, s_1)	(r_0, s_0)	(r_1, s_1)
(r_1, s_0)	(r_1, s_1)	(r_0, s_0)
(r_1, s_1)	(r_1, s_0)	(r_0, s_1)

- $q_0' = (r_0, s_0)$;
- $F' = \{(r_0, s_1)\}$.

Este autómato corresponde ao AFD produto obtido no Exemplo 1.32. A sua linguagem é L', o conjunto das palavras sobre $\{0,1\}$ que têm um número ímpar de 0's e um número par de 1's.

(ii) $D'' = (Q'', \{0,1\}, \delta'', q_0'', F'')$ em que

- $Q'' = \{p_0, p_1, p_2\}$;
- $\delta'' : Q'' \times \{0,1\} \to Q''$ é tal que

δ''	0	1
p_0	p_1	p_0
p_1	p_2	p_0
p_2	p_2	p_2

- $q_0'' = p_0$;
- $F'' = \{p_2\}$.

A linguagem de D'' é L'', o conjunto das palavras sobre $\{0,1\}$ que têm dois 0's consectivos.

(iii) $\overline{D''} = (\{p_0, p_1, p_2\}, \{0,1\}, \delta'', p_0, \{p_0, p_1\})$ em que δ'' é, naturalmente, a função de transição de D''.

(iv) $D = D' \times \overline{D''} = (Q, \{0,1\}, \delta, q_0, F)$ em que

– $Q = \{((r_0, s_0), p_0), ((r_0, s_1), p_0), ((r_1, s_0), p_0), ((r_1, s_1), p_0),$
$((r_0, s_0), p_1), ((r_0, s_1), p_1), ((r_1, s_0), p_1), ((r_1, s_1), p_1),$
$((r_0, s_0), p_2), ((r_0, s_1), p_2), ((r_1, s_0), p_2), ((r_1, s_1), p_2)\}$;

– $\delta : Q \times \{0, 1\} \to Q$ é tal que

δ	0	1
$((r_0, s_0), p_0)$	$((r_0, s_1), p_1)$	$((r_1, s_0), p_0)$
$((r_0, s_1), p_0)$	$((r_0, s_0), p_1)$	$((r_1, s_1), p_0)$
$((r_1, s_0), p_0)$	$((r_1, s_1), p_1)$	$((r_0, s_0), p_0)$
$((r_1, s_1), p_0)$	$((r_1, s_0), p_1)$	$((r_0, s_1), p_0)$
$((r_0, s_0), p_1)$	$((r_0, s_1), p_2)$	$((r_1, s_0), p_0)$
$((r_0, s_1), p_1)$	$((r_0, s_0), p_2)$	$((r_1, s_1), p_0)$
$((r_1, s_0), p_1)$	$((r_1, s_1), p_2)$	$((r_0, s_0), p_0)$
$((r_1, s_1), p_1)$	$((r_1, s_0), p_2)$	$((r_0, s_1), p_0)$
$((r_0, s_0), p_2)$	$((r_0, s_1), p_2)$	$((r_1, s_0), p_2)$
$((r_0, s_1), p_2)$	$((r_0, s_0), p_2)$	$((r_1, s_1), p_2)$
$((r_1, s_0), p_2)$	$((r_1, s_1), p_2)$	$((r_0, s_0), p_2)$
$((r_1, s_1), p_2)$	$((r_1, s_0), p_2)$	$((r_0, s_1), p_2)$

– $q_0 = ((r_0, s_0), p_0)$;

– $F = \{((r_0, s_1), p_0), ((r_0, s_1), p_1)\}$.

A linguagem deste autómato é o conjunto L, o conjunto das palavras sobre $\{0, 1\}$ que têm um número ímpar de 0's e um número par de 1's com a excepção das que têm dois 0's consecutivos. ▲

Exemplo 1.39 Seja L a linguagem sobre o alfabeto $\{a, b\}$ constituída pelas palavras que verificam pelo menos um dos seguintes requisitos

– começam e terminam em a;

– têm dois b's consecutivos.

Por exemplo, $aba \in L$ e $bbab \in L$ e $ababba \in L$ mas $abab \notin L$ (porque não tem dois b's consecutivos e, embora comece em a, não termina em a) e $baab$ (não tem dois b's consecutivos e, em particular, não começa em a).

Pretende-se construir um AFD D para a linguagem L. Note-se que designando por L' o conjunto das palavras sobre $\{a, b\}$ que começam e terminam em a e por L'' o conjunto das palavras sobre o mesmo alfabeto que têm dois b's consecutivos, tem-se que $L = L' \cup L''$. Mas $L' \cup L'' = \{a, b\}^* \backslash ((\{a, b\}^* \backslash L') \cap (\{a, b\}^* \backslash L''))$ e portanto, usando um vez mais o produto e a complementação de AFDS, pode obter-se um AFD para L da seguinte forma: (i) constrói-se um AFD D' para L'; (ii) constrói-se um AFD D'' para L''; (iii) constrói-se $\overline{D'}$; (iv) constrói-se $\overline{D''}$; (v) constrói-se $\overline{D'} \times \overline{D''}$; (vi) constrói-se $\overline{\overline{D'} \times \overline{D''}}$.

(i) $D' = (Q', \{a, b\}, \delta', q'_0, F')$ em que

– $Q' = \{p, q, r\}$;

– $\delta' : Q' \times \{a, b\} \to Q'$ é tal que

δ'	a	b
p	q	
q	q	r
r	q	r

– $q'_0 = p$;

– $F = \{q\}$.

Este autómato corresponde ao AFD construído no Exemplo 1.2 e a sua linguagem é L', o conjunto das palavras sobre $\{a, b\}$ que começam e terminam em a.

(ii) $D'' = (Q'', \{a, b\}, \delta'', q''_0, F'')$ em que

– $Q'' = \{s, t, u\}$;

– $\delta'' : Q'' \times \{a, b\} \to Q''$ é tal que

δ''	a	b
s	s	t
t	s	u
u	u	u

– $q''_0 = s$;

– $F'' = \{u\}$.

A linguagem de D'' é L'', o conjunto das palavras sobre $\{a, b\}$ que têm dois b's consectivos.

(iii) $TOT(D') = (\{p, q, r, q_\mathrm{t}\}, \{a, b\}, \delta'_\mathrm{t}, p, \{q\})$ em que

– $\delta'_\mathrm{t} : \{p, q, r, q_\mathrm{t}\} \times \{a, b\} \to \{p, q, r, q_\mathrm{t}\}'$ é tal que

δ'_t	a	b
p	q	q_t
q	q	r
r	q	r
q_t	q_t	q_t

pelo que $\overline{D'} = (\{p, q, r, q_\mathrm{t}\}, \{a, b\}, \delta'_\mathrm{t}, p, \{p, r, q_\mathrm{t}\})$.

(iv) $\overline{D''} = (\{s, t, u\}, \{a, b\}, \delta'', s, \{s, t\})$ em que δ'' é a função de transição de D''.

(v) $\overline{D'} \times \overline{D''} = (Q, \{a, b\}, \delta, (p, s), \{(p, s), (r, s), (q_\mathrm{t}, s), (p, t), (r, t), (q_\mathrm{t}, t)\})$ em que

– $Q = \{(p, s), (q, s), (r, s), (q_\mathrm{t}, s), (p, t), (q, t), (r, t), (q_\mathrm{t}, t)$
$(p, u), (q, u), (r, u), (q_\mathrm{t}, u)\}$;

– $\delta : Q \times \{a, b\} \to Q$ é tal que

δ	a	b
(p, s)	(q, s)	(q_{t}, t)
(q, s)	(q, s)	(r, t)
(r, s)	(q, s)	(r, t)
(q_{t}, s)	(q_{t}, s)	(q_{t}, t)
(p, t)	(q, s)	(q_{t}, u)
(q, t)	(q, s)	(r, u)
(r, t)	(q, s)	(r, u)
(q_{t}, t)	(q_{t}, s)	(q_{t}, u)
(p, u)	(q, u)	(q_{t}, u)
(q, u)	(q, u)	(r, u)
(r, u)	(q, u)	(r, u)
(q_{t}, u)	(q_{t}, u)	(q_{t}, u)

(vi) $D = \overline{\overline{D'} \times \overline{D''}} = (Q, \{a, b\}, \delta, (p, s), \{(q, s), (p, t), (q, t), (p, u), (q, u), (r, u), (q_{\mathrm{t}}, u)\})$ em que Q e δ são como em (v).

A linguagem deste autómato é o conjunto L, o conjunto das palavras sobre $\{a, b\}$ que ou começam por a e terminam em a ou têm dois b's consecutivos. ▲

Relativamente à utilização do produto e da complementação para construção modular de AFDS, no sentido acima referido, é importante observar o seguinte. No Exemplo 1.32, a partir de um AFD para a linguagem sobre $\{0, 1\}$ constituída pelas palavras com um número par de 1's e de um AFD para a linguagem sobre o mesmo alfabeto constituída pelas palavras com um número ímpar de 0's, obteve-se, usando o AFD produto, um AFD para a linguagem constituída pelas palavras que verificam ambos os requisitos, um número ímpar de 0's e um número par de 1's. O autómato assim obtido coincide com o AFD para esta linguagem construído no Exemplo 1.2, a menos do nome dos estados e, como adiante se verificará, este AFD é mesmo o AFD com o menor número de estados que é possível construir para esta linguagem. Mas nem sempre isto acontece, isto é, usando o AFD produto nem sempre se obtém um AFD para a linguagem em causa com o menor número possível de estados, nem mesmo que os AFDS usados como argumentos tenham eles próprios o menor número possível de estados. Por sua vez, o AFD D construído no Exemplo 1.39 recorrendo ao produto e complementação, não é o AFD com o menor número possível de estados para a linguagem L aí considerada pois o estado (p, t), por exemplo, não é acessível. Mas tal não impede que estas construções sejam úteis para obter AFDS para linguagens mais complicadas a partir de AFDS para linguagens mais simples, pois, como se estudará adiante, existem algoritmos de minimização que podem ser usados para, a partir de cada AFD assim construído, obter um AFD com o menor número possível de estados para a linguagem em causa.

1.5 Autómatos equivalentes

Quando uma linguagem é reconhecida por um certo AFD D, este autómato nunca é o único que reconhece esta linguagem. Existem muitos outros AFDS, na verdade mesmo um número infinito de AFDS, cuja linguagem é L_D. Basta pensar que, por exemplo, se podem sempre adicionar mais estados. Se dois AFDS reconhecem a mesma linguagem, estes AFDS dizem-se equivalentes.

Já tinha sido referido anteriormente que se num AFD existem estados inúteis, em geral, estes podem ser removidos obtendo-se um AFD que reconhece a mesma linguagem, ou seja, um AFD equivalente ao primeiro. Mas nem sempre só os estados inúteis podem ser removidos. Existem AFDS sem estados inúteis que, ainda assim, têm estados que não são necessários, sendo possível obter AFDS equivalentes aos de partida colapsando certos estados num só. Nesta secção mostra-se como se podem identificar tais estados. Os métodos desenvolvidos para identificar estes casos vão permitir ainda determinar se dois AFDS dados são ou não equivalentes.

É também importante saber se dado um AFD para uma linguagem existirá ou não um AFD equivalente com menos estados e, se existe, como se pode obtê-lo. Esta questão vai envolver os métodos abordados nesta secção, mas é tratada na secção 1.6.

Começa-se então pela definição de equivalência de AFDS.

Definição 1.40 AFDS EQUIVALENTES
Dois AFDS D_1 e D_2 são *equivalentes* se $L_{D_1} = L_{D_2}$. ◀

Seguem-se exemplos de autómatos equivalentes.

Exemplo 1.41 Os AFDS apresentados nos Exemplos 1.2 e 1.25 são AFDS equivalentes. Também são equivalentes os AFDS apresentados nos Exemplos 1.3 e 1.26. ▲

Os AFDS referidos no exemplo anterior correspondem a casos em que um dos autómatos tem estados inúteis. O exemplo seguinte apresenta dois AFDS equivalentes em que nenhum deles tem estados inúteis.

Exemplo 1.42 Considere-se o AFD $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{p, q, r, s\}$;
- $I = \{a, b\}$;
- $q_0 = p$;
- $\delta : Q \times I \to Q$ é tal que

δ	a	b
p	q	
q	q	r
r	q	s
s	q	r

- $F = \{q\}$.

A linguagem reconhecida por este AFD é o conjunto das palvras sobre o alfabeto $\{a, b\}$ que começam e terminam em a. Este autómato é assim equivalente ao apresentado no Exemplo 1.2, tendo, no entanto, mais um estado. Mas todos os estados deste autómato são acessíveis e produtivos, sendo por isso estados úteis. A questão neste caso está relacionada com o facto de não ser necessário estarem presentes simultaneamente os estados r e s. Se os estados s e r forem colapsados num só que se continue a designar por r e se considerar $\delta(r, b) = r$, obtém-se um AFD, mais precisamente o AFD do Exemplo 1.2, que é equivalente a D. Alternativamente, pode designar-se por s o novo estado mas haverá agora que considerar $\delta(s, b) = s$ e $\delta(q, b) = s$. A forma de identificar, estados com estas características, ditos estados equivalentes, é estudada adiante nesta secção. ▲

Enuncia-se em seguida a propriedade de que, em geral, podem ser removidos todos os estados inúteis de um AFD, bem como todas as transições de e para estes estados, obtendo-se um AFD equivalente. A única excepção é o caso dos AFDS vacuosos, caso em que todos os estados inúteis podem ser removidos, excepto o estado inicial.

Começa-se por definir o AFD $UT(D)$ o qual resulta do AFD D por eliminação dos seus estados inúteis. $UT(D)$ e D são equivalentes.

Definição 1.43 AFD $UT(D)$
Dado um AFD $D = (Q, I, \delta, q_0, F)$, o AFD $UT(D)$ é:

- se D não é vacuoso, $UT(D) = (Q', I, \delta', q_0, F')$ em que
  - $Q' = Q \backslash In_D$;
  - $\delta' : Q' \times I \to Q'$ é tal que

$$\delta'(q, a) = \begin{cases} \delta(q, a) & \text{se } \delta(q, a) \in F' \\ \\ \text{não def} & \text{se } \delta(q, a) \notin F' \end{cases}$$

    para cada $q \in Q'$ e $a \in I$;
  - $F' = F \backslash In_D$;
- se D é vacuoso, $UT(D)$ é o AFD vacuoso canónico com alfabeto I e estado inicial q_0. ◀

Note-se que $UT(D) = D$ se D náo tiver estados inúteis. Se D é um AFD vacuoso canónico então também $UT(D) = D$. O exemplo seguinte ilustra a construção de $UT(D)$.

Exemplo 1.44 Seja D o AFD apresentado no Exemplo 1.25. Então $UT(D) = (Q', I, \delta', q_0, F')$ em que

- $Q' = \{p, q, r\}$;

- $I = \{a, b\}$;
- $\delta' : Q' \times I \to Q'$ é tal que

δ	a	b
p	q	
q	q	r
r	q	r

- $q_0 = p$;
- $F = \{q\}$.

Removeu-se o único estado inútil de D, o estado s. ▲

Exemplo 1.45 Seja D o AFD apresentado no Exemplo 1.4. Como $In_D = \emptyset$, $UT(D) = D$. ▲

Proposição 1.46 Os AFDS D e $UT(D)$ são equivalentes. ■

Como já foi referido, existem AFDS em que mesmo estados acessíveis e produtivos são desnecessários. No Exemplo 1.42 obtém-se um AFD equivalente ao AFD D apresentado quando se colapsa o estado s com o estado r e se substitui a transição de r para s por uma transição para r para r. As palavras que estão associadas aos caminhos de D que começam em s e terminam no estado final, q, são exactamente as mesmas que estão associadas a caminhos que começam em r e terminam no estado final, ou seja, para cada palavra w sobre o alfabeto, $\delta^*(s, w) \in F$ se e só se $\delta^*(r, w) \in F$. Os estados que verificam esta propriedade são designados estados equivalentes. Estados que não sejam equivalentes dizem-se distinguíveis.

Definição 1.47 ESTADOS EQUIVALENTES E ESTADOS DISTINGUÍVEIS
Diz-se que dois estados p e q de um AFD $D = (Q, I, \delta, q_0, F)$ são

- *equivalentes* se, para cada $w \in I^*$, $\delta^*(p, w) \in F$ se e só se $\delta^*(q, w) \in F$;
- *distinguíveis* se não são estados equivalentes. ◀

Os estados p e q são distinguíveis precisamente quando existe uma palavra $w \in I^*$ que os distingue, isto é, tal que $\delta^*(p, w) \in F$ e $\delta^*(q, w) \notin F$ ou vice-versa. Em particular, se p é estado final e q não é estado final, p e q são estados distinguíveis, pois a palavra ϵ distingue-os, dado que $\delta^*(p, \epsilon) = p \in F$ e $\delta^*(q, \epsilon) = q \notin F$.

Se p é um estado é produtivo e q não é estado produtivo então estes dois estados são também distinguíveis, pois garante-se a existência de uma palavra $w \in I^*$ tal que $\delta^*(q, w) \in F$, mas sabe-se que $\delta^*(q, w) \notin F$.

Uma outra situação que permite identificar estados p e q distinguíveis é o caso em que existe um símbolo $a \in I$ tal que $\delta(p, a) \in Prd_D$ e $\delta(q, a)\uparrow$. Com efeito, $\delta^*(\delta(p, a), w) \in F$ para alguma palavra w, pelo que $\delta^*(p, a.w) \in F$. Por outro lado tem-se $\delta^*(q, a.w)\uparrow$. Assim a palavra $a.w$ distingue os estados p e q.

Observe-se ainda que, se p e q forem estados distinguíveis e $\delta(p', a) = p$ e $\delta(q', a) = q$, para algum símbolo a do alfabeto, então p' e q' são também estados distinguíveis. Com efeito, existe uma palavra w que distingue p e q. Suponhamos que se tem o caso em que $\delta^*(p, w)$ é estado final mas $\delta^*(q, w)$ não é, ou seja, $\delta^*(\delta(p', a), w)$ é estado final mas $\delta^*(\delta(q', a), w)$ não. Então $\delta^*(p', a.w)$ é estado final mas $\delta^*(q', a.w)$ não é, portanto a palavra $a.w$ distingue p' e q'.

Relativamente a estados equivalentes, tem-se que cada estado de um AFD é, evidentemente, equivalente a si próprio. No entanto, a identificação directa de estados equivalentes distintos é, em geral, mais difícil do que a identificação de estados distinguíveis. Para determinar se dois estados são distinguíveis basta encontrar uma palavra que os distinga. A prova da equivalência, por seu lado, pressupõe a verificação de uma condição para todas as palavras sobre o alfabeto, as quais, a menos que o alfabeto seja vazio, são em número infinito.

Observe-se, no entanto, que o conjunto de todos os pares de estados equivalentes de um AFD é precisamente o complementar do conjunto de todos os pares de estados distinguíveis, relativamente ao conjunto de todos os pares de estados do AFD. Assim, uma forma de identificar os estados equivalentes distintos, é começar por calcular o conjunto de todos os pares de estados distinguíveis do AFD. Se dois estados distintos não pertencerem a esse conjunto então são necessariamente equivalentes.

Exemplo 1.48 Considere-se o AFD apresentado no Exemplo 1.42. Os estados r e s são equivalentes, mas todos os outros pares de estados (distintos) são distinguíveis. Por exemplo, os estados p e q são distinguíveis porque p não é estado final, mas q é estado final. Os estados p e r são distinguíveis porque $\delta(r, b) = s$ e s é produtivo mas $\delta(p, b)\uparrow$. ▲

Exemplo 1.49 Considere-se o AFD $D = (Q, I, \delta, q_0, F)$ em que

- $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$;
- $I = \{a, b, c\}$;
- $\delta : Q \times I \to Q$ é tal que

δ	a	b	c
q_0	q_1	q_2	
q_1	q_1	q_0	q_4
q_2	q_2	q_0	q_5
q_3	q_3	q_1	q_4
q_4	q_2	q_5	
q_5	q_1	q_4	

- $F = \{q_4, q_5\}$.

Os estados q_1 e q_2 são equivalentes Os estados q_4 e q_5 são também equivalentes. Todos os outros pares de estados distintos são distinguíveis. Por exemplo, os estados q_0 e q_4 são distinguíveis porque q_0 não é estado final, mas q_4 é estado final. Os estados q_0 e q_1 são distinguíveis porque $\delta(q_1, c) = q_4$ e q_4 é produtivo mas $\delta(q_0, c)\uparrow$. Os estados q_1 e q_3 são distinguíveis porque $\delta^*(q_1, bc)\uparrow$ e $\delta^*(q_3, bc) = q_4 \in F$. ■

Exemplo 1.50 Nos AFDS apresentados nos Exemplos 1.2, 1.3, 1.4 e 1.5, todos os pares de estados distintos são distinguíveis. ▲

Quando num AFD se encontram dois estados distintos equivalentes, isso significa que o AFD pode ser simplificado, no sentido em que é possível colapsá-los e obter um AFD equivalente. As transições para o estado que deixa de existir são substituídas por transições para o estado equivalente cuja designação é mantida. Na proposição seguinte enuncia-se esta propriedade dos AFDS.

Proposição 1.51 Seja $D = (Q, I, \delta, q_0, F)$ um AFD e sejam $p, q \in Q$ estados distintos e equivalentes com $p \neq q_0$. O AFD $D' = (Q', I, \delta', q_0, F')$ em que

- $Q' = Q \backslash \{p\}$;
- $\delta' : Q' \times I \to Q'$ é tal que

$$\delta'(q', a) = \begin{cases} \delta(q', a) & \text{se } \delta(q', a) \in Q' \\ q & \text{se } \delta(q', a) = p \\ \text{não def} & \text{se } \delta(q', a)\uparrow \end{cases}$$

  para cada $q' \in Q'$ e $a \in I$;
- $F' = F \backslash \{p\}$;

é equivalente a D. ■

Note-se que o requisito $p \neq q_0$ no enunciado da proposição anterior não implica perda de generalidade pois, como existe um único estado inicial, dados dois estados distintos um deles será necessariamente não inicial.

Apresenta-se seguidamente um exemplo de aplicação do resultado estabelecido nesta proposição.

Exemplo 1.52 Considere-se o AFD D apresentdo no Exemplo 1.49. Como foi aí referido os estados q_1 e q_2 são equivalentes. Pode então colapsar-se estes estes dois estados num só. Suponha-se que o estado resultante do colapso se continua a designar q_1. O autómato equivalente que se obtém é $D' = (Q', I, \delta', q_0, F')$ em que

- $Q = \{q_0, q_1, q_3, q_4, q_5\}$;
- $I = \{a, b, c\}$;
- $\delta' : Q' \times I \to Q'$ é tal que

δ	a	b	c
q_0	q_1	q_1	
q_1	q_1	q_0	q_4
q_3	q_3	q_1	q_4
q_4	q_1	q_5	
q_5	q_1	q_4	

- $F = \{q_4, q_5\}$.

Note-se as transições para q_2 foram substituídas por transições para q_1.

O AFD D' tem um outro par de estados equivalentes, os estados q_4 e q_5. Supondo que o estado resultante do coplapso destes dois se continua a designar q_4, obtém-se o AFD $D'' = (Q'', I, \delta'', q_0, F'')$ em que

- $Q = \{q_0, q_1, q_3, q_4\}$;
- $I = \{a, b, c\}$;
- $\delta' : Q' \times I \to Q'$ é tal que

δ	a	b	c
q_0	q_1	q_1	
q_1	q_1	q_0	q_4
q_3	q_3	q_1	q_4
q_4	q_1	q_4	

- $F = \{q_4\}$.

As trnasições para q_5 foram substituídas por transições para q_4.

As linguagens L_D, $L_{D'}$ e $L_{D''}$ são iguais. Referia-se que em todos estes AFDS há um estado inútil: o estado q_3. Com efeito, este estado não é acessível ▲

Num AFD pode acontecer que dois estados p e q sejam equivalentes e os estados q e r também, ou seja, existem dois pares de estados equivalentes que têm um estado em comum. Tendo em conta a Definição 1.47, é fácil perceber que p e r são também estados equivalentes. Neste caso, os três estados podem ser colapsados num só, obtendo-se ainda um AFD equivalente ao inicial se as transições forem convenientemente modificadas. O resultado enunciado na Proposição 1.51 pode assim ser estendido a conjuntos $C \subseteq Q$ tais que quaisquer dois estados em C são equivalentes. Neste caso colapsam-se todos os estados de C num só. Deixa-se como exercício enunciar a extensão do resultado referido na Proposição 1.51 ao caso em que se colapsam mais de dois estados.

Como referido anteriormente, a identificação directa de estados equivalentes distintos é, em geral, mais difícil do que a identificação de estados distinguíveis e, dada a complementaridade do conjunto de todos os pares de estados equivalentes de um AFD face ao de todos os pares de estados distinguíveis, relativamente ao conjunto de todos os pares de estados do AFD, uma maneira de identificar estados equivalentes é começar por identificar todos os pares de estados distinguíveis. Apresenta-se na sequência um algoritmo que calcula o conjunto de todos os pares de estados distinguíveis de um AFD.

Para facilitar a exposição, introduzem-se as notações seguintes. Usa-se uma notação para representar pares *não ordenados de estados* ou, mais simplesmente, pares de estados: denota-se indiferentemente por $[p, q]$ ou $[q, p]$ o par constituído pelo estados distintos p e q. Note-se que, neste sentido, um par de estados não é mais do que um conjunto constituído por dois estados. Denota-se por Prs_D o conjunto de todos os pares de estados de D, por Eqv_D o conjunto de todos os $[p, q] \in Prs_D$ tais que p e q são estados equivalentes e por Dst_D o conjunto de todos os $[p, q] \in Prs_D$ tais que p e q são estados distinguíveis.

Apresenta-se agora o algoritmo de procura de estados distinguíveis, APED, que calcula o conjunto de todos os pares de estados distinguíveis do AFD de entrada D.

ALGORITMO DE PROCURA DE ESTADOS DISTINGUÍVEIS (APED):

ENTRADA: AFD $D = (Q, I, \delta, q_0, F)$
SAÍDA: conjunto $APED(D)$

1. $\Delta \quad := \{[p,q] : p \in F \text{ e } q \notin F\} \cup$
   $\{[p,q] : p \in Prd_D \text{ e } q \notin Prd_D\} \cup$
   $\{[p,q] : \text{existe } a \in I \text{ tal que } \delta(p,a) \in Prd_D \text{ e } \delta(q,a)\uparrow\}$;
2. $Aux := \{[p,q] \notin \Delta : \text{existe } a \in I \text{ tal que } [\delta(p,a), \delta(q,a)] \in \Delta\}$;
3. enquanto $\Delta \cup Aux \neq Prs_D$ e $Aux \neq \emptyset$
   3.1. $\Delta := \Delta \cup Aux$;
   3.2. $Aux := \{[p,q] \notin \Delta : \text{ existe } a \in I \text{ tal que } [\delta(p,a), \delta(q,a)] \in Aux\}$;
4. $APED(D) := \Delta$.

O APED começa por identificar pares de estados em que um dos estados é final e outro não é. Identifica também pares estados em que um é produtivo e outro não. Na fase inicial identifica ainda pares de estados tais que existe uma transição associada a um símbolo do alfabeto partindo de um deles para um estado produtivo e não existe transição associada a esse símbolo partindo do outro. Observe-se que todos os pares de estados nas condições acima referidas são pares de estados distinguíveis.

De seguida, vão identificar-se novos pares de estados. Procuram-se estados distintos p e q tais que haja uma transição associada a um dado símbolo a partir de p e haja também uma transição associada ao mesmo símbolo a partir de q, sendo que essas transições têm de ser para estados que já tenham sido identificados previamente pelo APED. Note-se que, sendo estes estados distinguíveis, os estados p e q são também distinguíveis.

O processo repete-se até já todos os elementos de Prs_D, terem sido identificados pelo APED, ou até não se conseguirem encontrar novos pares de estados distinguíveis, isto é, aplicando o processo de identificação de novos pares referido no parágrafo anterior aos pares identificados até ao momento, só se encontram pares de estados que já tenham sido identificados previamente pelo APED.

A execução do APED termina sempre porque, como existe um número finito de estados, existem também um número finito de pares de estados e, em particular, um número finito de pares de estados distinguíveis. Garante-se também que, quando termina a sua execução, só foram identificados pares de estados distinguíveis e foram identificados todos os pares nessas condições, isto é, garante-se que $APED(D) = Dst_D$. Estas propriedades de APED são enunciadas na Proposição 1.56.

Apresentam-se seguidamente vários exemplos de utilização do APED.

Exemplo 1.53 Considere-se o AFD apresentado no Exemplo 1.42. Para encontrar todos os pares de estados distinguíveis de D seguindo o APED, há que

calcular o valor inicial de Δ como indicado na instrução 1. Para tal começa-se por considerar todos os pares de estados em que um seja final e outro não. São eles

$$[p, q],\ [q, s],\ [q, r]$$

De seguida consideram-se os pares de estados em que um estado é produtivo e outro não. Neste caso não existem tais pares porque todos os estados são produtivos. Por fim, procuram-se pares de estados tais que a partir de um deles exista uma transição associada a um símbolo para um estado produtivo, e a partir do outro não exista transição associada a este símbolo. Neste caso encontram-se os pares

$$[p, q],\ [p, r],\ [p, s]$$

Em particular, observe-se que $\delta(p, b)\uparrow$ e $\delta(r, b) = s$. Assim, o valor de Δ calculado pela instrução 1 é $\Delta = \{[p, q], [q, s], [q, r], [p, r], [p, s]\}$.

Há agora que verificar se a partir dos estados distinguíveis encontrados até ao momento se encontram novos pares distinguíveis, isto é, há que calcular o valor inicial de Aux, como indicado na instrução 2 do APED. Por exemplo, dado que o par $[p, q] \in \Delta$, há que procurar estados q' e q'' e um símbolo i do alfabeto tais que $\delta(q', i) = p$ e $\delta(q'', i) = q$. Neste caso, não existem q' e q'' nessas condições, pelo que o par $[p, q]$ não permite encontrar mais pares de estados distinguíveis. Procedendo de modo semelhante para os outros pares em Δ, não se encontram mais pares de estados distingíveis. Isto significa que o valor que é atribuído a Aux na instrução 2 é o conjunto vazio.

Dado que $Aux = \emptyset$, a guarda do ciclo correspondente à instrução 3 é falsa. É então executada a instrução 4 que atribui a $APED(D)$ o valor corrente de Δ. Assim, o conjunto dos pares de estados calculado pelo APED é

$$APED(D) = \{[p, q], [q, s], [q, r], [p, r], [p, s]\}$$

que, tendo em conta a propriedade do APED enunciada, é de facto Dst_D, o conjunto de todos os pares de estados distinguíves deste AFD. Consequentemente,

$$Eqv_D = \{[r, s]\}$$

é o conjunto de pares de estados equivalentesde de D.

Ao executar manualmente este algoritmo, para faciltar, pode utilizar-se uma tabela que se vai preenchendo à medida que se vão identificando pares de estados distinguíveis. A tabela, no caso deste AFD, é

q			
r			
s			
	p	q	r

sendo construída por forma a permitir referenciar cada par de estados de D. Note-se que nesta tabela existe uma linha para cada estado, com a excepção do estado p, e existe uma coluna para cada estado, com a excepção do estado s.

Durante o cálculo do valor inicial de Δ, instrução 1, vão-se assinalando na tabela os pares que vão sendo identificados, obtendo-se após a execução desta instrução:

q	$\times$		
r	$\times$	$\times$	
s	$\times$	$\times$	
	p	q	r

O cálculo do valor inicial de Aux, instrução 2, implica considerar cada par em Δ e verificar se a partir dele se conseguem identificar novos pares de estados. Após a análise de cada par em Δ, o par é referenciado na tabela usando $\otimes$, por exemplo, e novos pares de estados assim identificados (se existirem) são assinalados na tabela com $\times$, como anteriormente. Começando pelo par $[p, q]$, por exemplo, conclui-se que nenhum novo par é identificado, pelo que a tabela resultante é

q	$\otimes$		
r	$\times$	$\times$	
s	$\times$	$\times$	
	p	q	r

Prosseguindo de modo análogo para os restantes casos, não se encontram mais pares de estados distingíveis e portanto, após a execução da instrução 2, a tabela correspondente é

q	$\otimes$		
r	$\otimes$	$\otimes$	
s	$\otimes$	$\otimes$	
	p	q	r

Como se viu, neste caso, o corpo do ciclo correspondente à instrução 3 não é executado e portanto não há identificação de mais estados. A tabela final é a acima apresentada. Nesta tabela estão referenciados todos os pares de estados identificados pelo algoritmo e também a indicação de que todos eles foram devidamente analisados para se determinar se novos pares se poderiam identificar a partir deles. Naturalmente, no caso de serem executadas uma mais iterações do ciclo, a forma de continuar a preencher a tabela seria idêntica à acima referida. ▲

Exemplo 1.54 Considere-se o AFD apresentado no Exemplo 1.49. O objectivo é novamente encontrar todos os pares de estados distinguíveis de D seguindo o APED. Note-se que todos os estados são produtivos. Faz-se agora uma descrição do cálculo mais sucinta do que a apresentada no exemplo anterior e, em simutâneo, vai-se apresentando também a construção da tabela auxiliar referida.

Os pares de estados distinguíveis encontrados na fase inicial, isto é, os pares encontrados ao calcular o valor de Δ executando a instrução 1 do APED, são os indicados na tabela seguinte:

q_1	$\times$				
q_2	$\times$				
q_3	$\times$				
q_4	$\times$	$\times$	$\times$	$\times$	
q_5	$\times$	$\times$	$\times$	$\times$	
	q_0	q_1	q_2	q_3	q_4

Por exemplo, q_0 e q_4 são estados distinguíveis porque q_0 não é estado final e q_4 é. Por outro lado, q_0 e q_1 são estados distinguíveis porque a partir de q_0 não existe transição associada a c, mas a partir de q_1 existe transição associada a c para o estado q_4 que é produtivo.

A partir dos pares já encontrados (guardados em Δ) tentam-se encontrar novos pares como indica a instrução 2 do APED. Neste caso, considerando o par $[q_0, q_1]$ tem-se $\delta(q_1, b) = q_0$ e $\delta(q_3, b) = q_1$, logo identifica-se um novo par: $[q_1, q_3]$. De modo análogo se identifica também $[q_2, q_3]$. Mais nenhum novo par é identificado e portanto, após a execução da instrução 2, $Aux = \{[q_1, q_3], [q_2, q_3]\}$.

Como $\Delta \cup Aux$ ainda não é o conjunto de todos os pares de estados de D e $Aux \neq \emptyset$, a guarda do ciclo correspondente à instrução 3 é verdadeira e as instruções 3.1 e 3.2 serão executadas. O valor de Δ é actualizado com o valor corrente de Aux. Seguindo as convenções adoptadas, tal corresponde a actualizar a tabela obtendo-se

q_1	$\otimes$				
q_2	$\otimes$				
q_3	$\otimes$	$\times$	$\times$		
q_4	$\otimes$	$\otimes$	$\otimes$	$\otimes$	
q_5	$\otimes$	$\otimes$	$\otimes$	$\otimes$	
	q_0	q_1	q_2	q_3	q_4

A execução da instrução 3.2 corresponde a actualizar o valor de Aux como indicado, e portanto o objectivo é agora considerar os pares $[q_1, q_3]$ e $[q_2, q_3]$ e tentar a partir deles identificar mais novos pares de estados distinguíveis. Neste caso não são identificados mais pares. pelo que o novo valor de Aux é o conjunto vazio. A tabela resultante é

q_1	$\otimes$				
q_2	$\otimes$				
q_3	$\otimes$	$\otimes$	$\otimes$		
q_4	$\otimes$	$\otimes$	$\otimes$	$\otimes$	
q_5	$\otimes$	$\otimes$	$\otimes$	$\otimes$	
	q_0	q_1	q_2	q_3	q_4

A guarda do ciclo é agora falsa, pois $Aux = \emptyset$, pelo que se segue a execução da instrução 4 de que resulta

$$APED(D) = \{[q_0, q_1], [q_0, q_2]], [q_0, q_3], [q_0, q_4], [q_0, q_5], [q_1, q_3], [q_1, q_4] \\ [q_1, q_5], [q_2, q_3], [q_2, q_4], [q_2, q_5], [q_3, q_4], [q_3, q_5]\}.$$

Tendo em conta a propriedade do APED enunciada, este conjunto é precisamente o conjunto de todos os pares de estados distinguíveis de D. Consequentemente

$$Eqv_D = \{[q_1, q_2], [q_4, q_5]\}$$

é o conjunto de pares de estados equivalentesde de D. ▲

Exemplo 1.55 Considere-se o AFD apresentado no Exemplo 1.4. De novo se pretende encontrar todos os pares de estados distinguíveis de D seguindo o APED. Note-se que todos os estados são produtivos.

Os pares de estados distinguíveis encontrados na fase inicial (valor inicial de Δ, instrução 1) são os indicados na tabela:

P_0I_1			
I_0I_1			
I_0P_1	$\times$	$\times$	$\times$
	P_0P_1	P_0I_1	I_0I_1

A partir dos pares já encontrados, tentam-se encontrar novos pares como indica a instrução 2 do APED. Neste caso, considerando o par $[P_0P_1, I_0P_1]$, identifica-se um novo par: $[P_0I_1, I_0I_1]$. A partir do par $[P_0I_1, I_0P_1]$, identifica-se um novo par: $[P_0P_1, I_0I_1]$. Por último, a partir do par $[I_0I_1, I_0P_1]$, identifica-se um novo par: $[P_0P_1, P_0I_1]$. Deste modo, após a execução da instrução 2 tem-se $Aux = \{[P_0I_1, I_0I_1], [P_0P_1, I_0I_1], [P_0P_1, P_0I_1]\}$. A tabela fica:

P_0I_1	$\times$		
I_0I_1	$\times$	$\times$	
I_0P_1	$\otimes$	$\otimes$	$\otimes$
	P_0P_1	P_0I_1	I_0I_1

Neste momento $\Delta \cup Aux$ é já o conjunto de todos os pares de estados de D, pelo que a guarda do ciclo é falsa. Como é natural, nesta situação, não se prossegue a procura de mais pares de estados distinguíveis, pois todos os pares de estados do AFD foram identificados. Assim, $APED(D) = Dst_D = Prs_D$ e $Eqv_D = \emptyset$. ▲

Enunciam-se de seguida as propriedades de APED referidas anteriormente.

Proposição 1.56 A execução do APED termina sempre e, sendo D o AFD de entrada, $APED(D) = Dst_D$. ■

O APED pode ser usado para determinar se dois AFDS D_1 e D_2 são ou não equivalentes. Como se referiu anteriormente, pode assumir-se, sem perda de generalidade, que os conjuntos dos estados Q_1 e Q_2 dos dois autómatos são disjuntos. Assuma-se também, para já, que os AFDS têm o mesmo alfabeto. A ideia é construir a partir de D_1 e D_2 um novo AFD D, com o mesmo alfabeto, o qual pode ser visto como uma união destes AFDS, no seguinte sentido: o seu conjunto de estados é $Q_1 \cup Q_2$, mantêm-se as transições existentes em D_1 e as transições em D_2, e o seu conjunto de estados finais é $F = F_1 \cup F_2$, sendo F_1 o conjunto dos estados finais de D_1 e F_2 o de D_2. O estado inicial tanto pode ser o de D_1 como o de D_2. Como Q_1 e Q_2 são disjuntos e se mantêm as transições, os caminhos de D que começam no estado inicial de D_1, q_0^1, e terminam num estado em F são exactamente os caminhos de D_1 que começam em q_0^1 e terminam num estado em F_1, e as palavras associadas a estes caminhos são exactamente as mesmas. Note-se que o conjunto destas palavras é precisamente L_{D_1}. O mesmo raciocínio se pode fazer relativamente a caminhos de D que têm início no estado inicial de D_2, q_0^2, e terminam num estado final de D. Deste modo, para testar a equivalência de D_1 e D_2, basta determinar se o conjunto das palavras associadas a caminhos de D que começam em q_0^1 e terminam num estado em F é o conjunto das palavras associadas a caminhos de D que começam em q_0^2 e terminam num

estado em F. Isto corresponde precisamente a determinar se os estados q_0^1 e q_0^2 são ou não estados equivalentes de D. O algoritmo de teste à equivalência de AFDS, ATEQ, é assim o seguinte.

ALGORITMO DE TESTE À EQUIVALÊNCIA DE AFDS (ATEQ):

ENTRADA: AFDS $D_1 = (Q_1, I, \delta_1, q_0^1, F_1)$ e $D_2 = (Q_2, I, \delta_2, q_0^2, F_2)$
tais que $Q_1 \cap Q_2 = \emptyset$
SAÍDA: valor booleano $ATEQ(D_1, D_2)$

1. construir o AFD $D = (Q_1 \cup Q_2, I, \delta, q_0^1, F_1 \cup F_2)$ em que
$$\delta(q,a) = \begin{cases} \delta_1(q,a) & \text{se } q \in Q_1 \text{ e } \delta_1(q,a)\downarrow \\ \delta_2(q,a) & \text{se } q \in Q_2 \text{ e } \delta_2(q,a)\downarrow \\ \text{não def} & \text{caso contrário} \end{cases}$$
2. calcular $APED(D)$;
3. $ATEQ(D_1, D_2) =$ **true** se $[q_0^1, q_0^2] \notin APED(D)$ e
$ATEQ(D_1, D_2) =$ **false** caso contrário.

A execução do ATEQ termina sempre e identifica como equivalentes dois AFDS se e só se eles são equivalentes. Estas propriedades são enunciadas na Proposição 1.58.

Apresenta-se agora um exemplo de aplicação do ATEQ.

Exemplo 1.57 Considere-se o AFD $D_1 = (Q_1, I, \delta_1, q_0^1, F_2)$ em que

- $Q_1 = \{p_0, p_1, p_2, p_3, p_4, p_5\}$;
- $I = \{a, b, c\}$;
- $\delta_1 : Q_1 \times I \to Q_1$ é tal que

δ_1	a	b	c
p_0	p_1	p_2	
p_1	p_1	p_0	p_4
p_2	p_2	p_0	p_5
p_3	p_3	p_1	p_4
p_4	p_2	p_5	
p_5	p_1	p_4	

- $q_0^1 = p_0$;
- $F_1 = \{p_4, p_5\}$.

Considere-se também o AFD $D_2 = (Q_2, I, \delta_2, q_0^2, F_2)$ em que

- $Q_2 = \{r_0, r_1, r_2, r_3\}$;
- $I = \{a, b, c\}$;

- $\delta_2 : Q_2 \times I \to Q_2'$ é tal que

δ	a	b	c
r_0	r_1	r_1	
r_1	r_1	r_0	r_3
r_2	r_2	r_1	r_3
r_3	r_1	r_3	

- $q_0^2 = r_0$;
- $F_2 = \{r_3\}$.

O objectivo é usar o ATEQ para determinar se estes AFDS são ou não equivalentes.

O primeiro passo é construir o AFD indicado na instrução 1 do ATEQ. É o AFD $D = (Q, I, \delta, p_0, F)$ em que

- $Q = \{p_0, p_1, p_2, p_3, p_4, p_5, r_0, r_1, r_2, r_3\}$;
- $\delta : Q \times I \to Q$ é tal que

δ_1	a	b	c
p_0	p_1	p_2	
p_1	p_1	p_0	p_4
p_2	p_2	p_0	p_5
p_3	p_3	p_1	p_4
p_4	p_2	p_5	
p_5	p_1	p_4	
r_0	r_1	r_1	
r_1	r_1	r_0	r_3
r_2	r_2	r_1	r_3
r_3	r_1	r_3	

- $F = \{p_4, p_5, r_3\}$.

Há agora que calcular o conjunto dos pares de estados distinguíveis de D, usando o APED. Ilustram-se apenas sucintamente as diferentes fases da construção da tabela usual. Os pares de estados distinguíveis que primeiro são identificados são os indicados na tabela seguinte:

p_1	$\times$								
p_2	$\times$								
p_3	$\times$								
p_4	$\times$	$\times$	$\times$	$\times$					
p_5	$\times$	$\times$	$\times$	$\times$					
r_0		$\times$	$\times$	$\times$	$\times$	$\times$			
r_1	$\times$				$\times$	$\times$	$\times$		
r_2	$\times$				$\times$	$\times$	$\times$		
r_3	$\times$	$\times$	$\times$	$\times$			$\times$	$\times$	$\times$
	p_0	p_1	p_2	p_3	p_4	p_5	r_0	r_1	r_2

A partir de $[p_0, p_1]$ identificam-se $[p_1, p_3]$ e $[p_2, p_3]$. A partir de $[p_0, r_1]$ identificam-se $[p_1, r_2]$ e $[p_2, r_2]$. A partir de $[p_1, r_0]$ identifica-se $[p_3, r_1]$. A partir de $[r_0, r_1]$ identifica-se $[r_1, r_2]$. A tabela obtida é:

p_1	$\otimes$							
p_2	$\times$							
p_3	$\times$	$\times$	$\times$					
p_4	$\times$	$\times$	$\times$	$\times$				
p_5	$\times$	$\times$	$\times$	$\times$				
r_0		$\otimes$	$\times$	$\times$	$\times$	$\times$		
r_1	$\otimes$			$\times$	$\times$	$\times$	$\otimes$	
r_2	$\times$	$\times$	$\times$		$\times$	$\times$	$\times$	$\times$	
r_3	$\times$	$\times$	$\times$	$\times$			$\times$	$\times$	$\times$
	p_0	p_1	p_2	p_3	p_4	p_5	r_0	r_1	r_2

Dos outros pares identificados inicialmente não resultam novos pares:

p_1	$\otimes$							
p_2	$\otimes$							
p_3	$\otimes$	$\otimes$	$\times$					
p_4	$\otimes$	$\otimes$	$\otimes$	$\otimes$				
p_5	$\otimes$	$\otimes$	$\otimes$	$\times$				
r_0		$\otimes$	$\otimes$	$\otimes$	$\otimes$	$\otimes$		
r_1	$\otimes$			$\times$	$\otimes$	$\otimes$	$\otimes$	
r_2	$\otimes$	$\times$	$\times$		$\otimes$	$\otimes$	$\otimes$	$\times$	
r_3	$\otimes$	$\otimes$	$\otimes$	$\otimes$			$\otimes$	$\otimes$	$\otimes$
	p_0	p_1	p_2	p_3	p_4	p_5	r_0	r_1	r_2

A partir de $[p_1, p_3]$, $[p_2, p_3]$, $[p_1, r_2]$, $[p_2, r_2]$, $[p_3, r_1]$ e $[r_1, r_2]$ também não se identificam novos pares:

p_1	$\otimes$							
p_2	$\otimes$							
p_3	$\otimes$	$\otimes$	$\otimes$					
p_4	$\otimes$	$\otimes$	$\otimes$	$\otimes$				
p_5	$\otimes$	$\otimes$	$\otimes$	$\otimes$				
r_0		$\otimes$	$\otimes$	$\otimes$	$\otimes$	$\otimes$		
r_1	$\otimes$			$\otimes$	$\otimes$	$\otimes$	$\otimes$	
r_2	$\otimes$	$\otimes$	$\otimes$		$\otimes$	$\otimes$	$\otimes$	$\otimes$	
r_3	$\otimes$	$\otimes$	$\otimes$	$\otimes$			$\otimes$	$\otimes$	$\otimes$
	p_0	p_1	p_2	p_3	p_4	p_5	r_0	r_1	r_2

Identificados todos os pares de estados distinguíveis, conclui-se que p_0 e r_0 são estados equivalentes. Assim, os AFDS dados são equivalentes. ▲

Enunciam-se agora as propriedades do ATEQ acima referidas.

Proposição 1.58 A execução do ATEQ termina sempre e, sendo D_1 e D_2 os AFDS de entrada, $ATEQ(D_1, D_2) =$ **true** se e só se D_1 e D_2 são equivalentes. ■

Recorda-se que o ATEQ pressupõe que os AFDS D_1 e D_2 têm o mesmo alfabeto. É fácil perceber que, mesmo no caso de os alfabetos serem diferentes, é possível utilizar este algoritmo para determinar se os AFDS são equivalentes. Basta considerar AFDS semelhantes a D_1 e D_2 mas cujo alfabeto é a união dos alfabetos de D_1 e D_2 e a função de transição é convenientemente modidficada, à semelhança do que foi referido a propósito do cálculo do AFD produto no caso os AFDS de partida não terem o mesmo alfabeto.

1.6 Minimização

Nas secções anteriores foram estudadas situações em que é possível eliminar estados de um AFD e obter ainda um AFD equivalente. São AFDS que têm estados inúteis ou que têm estados equivalentes. Uma questão que se pode colocar é a de saber quando é que deixa de ser possível diminuir o número de estados sem alterar a linguagem reconhecida pelo AFD. É importante saber se o AFD dado é um AFD mínimo, isto é, ter a certeza de que não é possível construir um AFD equivalente ao dado mas com menos estados. Uma outra propriedade importante dos AFDS que será estudada é a seguinte: se dois AFDS (não vacuosos) são equivalentes e não têm estados inúteis nem estados distintos equivalentes, então eles são isomorfos. Isto significa que os AFDS são essencialmente o mesmo, a menos do nome dos respectivos estados. Uma consequência desta propriedade é o facto de se poder concluir que para cada linguagem regular existe, a menos de isomorfismo, um único AFD mínimo. Esta secção é dedicada ao estudo destas propriedades dos AFDS.

Começa-se por definir a noção de AFD mínimo. De seguida mostra-se como, dado um AFD D, é possível construir um AFD mínimo equivalente a D.

Definição 1.59 AFD MÍNIMO
Um AFD D diz-se *mínimo* se não existe nenhum AFD equivalente a D com menos estados do que D. ◀

Para construir um AFD mínimo equivalente a um AFD dado vai ser necessário considerar um certa partição do conjunto Q dos seus estados. Uma partição de Q é um conjunto de subconjuntos não vazios de Q, disjuntos dois a dois e cuja união é Q. A partição que é relevante para a construção de um AFD mínimo é a partição induzida pelos estados equivalentes do AFD.

Definição 1.60 PARTIÇÃO INDUZIDA PELOS ESTADOS EQUIVALENTES
Seja D um AFD com conjunto de estados Q. A *partição de Q induzida por Eqv_D* é o conjunto $\bigcup_{q\in Q}\{C[q]\}$ em que $C[q] = \{q\} \cup \{p \in Q : [p, q] \in Eqv_D\}$ para cada $q \in Q$. ◀

Deixa-se como exercício mostrar que a partição induzida por Eqv_D da Definição 1.60 é de facto uma partição de Q

Exemplo 1.61 Considere-se o autómato D referido no Exemplo 1.42. No Exemplo 1.53 conclui-se que $Eqv_D = \{[r, s]\}$. Seguindo a Definição 1.60

- $C[p] = \{p\}$;

- $C[q] = \{q\}$;
- $C[r] = C[s] = \{r, s\}$.

A partição induzida por Eqv_D é assim $\{\{p\}, \{q\}, \{r, s\}\}$. ▲

Exemplo 1.62 Considere-se o autómato D referido no Exemplo 1.52. No Exemplo 1.54 conclui-se que $Eqv_D = \{[q_1, q_2], [q_4, q_5]\}$. Seguindo a Definição 1.60

- $C[q_0] = \{q_0\}$;
- $C[q_3] = \{q_3\}$;
- $C[q_1] = C[q_2] = \{q_1, q_2\}$.
- $C[q_4] = C[q_5] = \{q_4, q_5\}$.

A partição induzida por Eqv_D é assim $\{\{q_0\}, \{q_3\}, \{q_1, q_2\}, \{q_4, q_5\}\}$. ▲

Exemplo 1.63 Considere-se o autómato D referido no Exemplo 1.4. No Exemplo 1.55 conclui-se que $Eqv_D = \emptyset$. Seguindo a Definição 1.60

- $C[\mathrm{P_0P_1}] = \{\mathrm{P_0P_1}\}$;
- $C[\mathrm{P_0I_1}] = \{\mathrm{P_0I_1}\}$;
- $C[\mathrm{I_0P_1}] = \{\mathrm{I_0P_1}\}$;
- $C[\mathrm{I_0I_1}] = \{\mathrm{I_0I_1}\}$.

A partição induzida por Eqv_D é assim $\{\{\mathrm{P_0P_1}\}, \{\mathrm{P_0I_1}\}, \{\mathrm{I_0P_1}\}, \{\mathrm{I_0I_1}\}\}$. ▲

A noção de equivalência entre estados permite estabelecer uma relação binária $\sim$ no conjunto Q dos estados de um AFD D: $p \sim q$ se p e q são estados equivalentes. Esta relação é reflexiva, simétrica e transitiva, logo é uma relação de equivalência. O conjunto das classes de equivalência desta relação, isto é, o conjunto quociente $Q/\sim$, é precisamente a partição de Q induzida por Eqv_D definida acima. Para cada estado $q \in Q$, o conjunto $C[q]$ referido na Definição 1.60 é a classe de equivalência de q determinada pela relação $\sim$.

São relevantes as seguintes propriedades da partição induzida por Eqv_D.

Proposição 1.64 Seja $D = (Q, I, \delta, q_0, F)$ um AFD e C um conjunto da partição induzida por Eqv_D.

1. Se em C existe um estado final de D então C só tem estados finais.
2. Se $\delta(q, a)\uparrow$ para algum $q \in C$ então $\delta(q', a)\uparrow$ para qualquer $q' \in C$.
3. Se $q \in C$ e $\delta(q, a) = p$ então, para qualquer $q' \in C$, $\delta(q', a) = p$ ou $\delta(q', a) = p'$ com $[p, p'] \in Eqv_D$. ■

Mostra-se agora como construir um AFD mínimo equivalente a um AFD dado.

Definição 1.65 MINIMIZAÇÃO DE AFD
Dado um AFD $D = (Q, I, \delta, q_0, F)$, a *minimização* de D é o AFD $MIN(D)$ construído como se segue:

- se D é vacuoso então $MIN(D)$ é o AFD vacuoso canónico com alfabeto I e estado inicial q_0;
- se D não é vacuoso, $MIN(D) = (Q_{\mathrm{m}}, I, \delta_{\mathrm{m}}, q_0^{\mathrm{m}}, F_{\mathrm{m}})$ em que
  - $Q_{\mathrm{m}} = \{C_0, C_1, \ldots, C_n\}$ é a partição do conjunto dos estados de $UT(D)$ induzida por $Eqv_{UT(D)}$, com $q_0 \in C_0$;
  - $\delta_{\mathrm{m}} : Q_{\mathrm{m}} \times I \to Q_{\mathrm{m}}$ é tal que

    $$\delta_{\mathrm{m}}(C_i, a) = \begin{cases} C_j & \text{se } \delta(q,a) \in C_j \text{ para algum } q \in C_i \\ \text{não def} & \text{se } \delta(q,a)\uparrow \text{ para algum } q \in C_i \end{cases}$$

    para cada $1 \leq i \leq n$ e $a \in I$
  - $q_0^{\mathrm{m}} = C_0$;
  - $F_{\mathrm{m}} = \{C_i : 1 \leq i \leq n \text{ e } C_i \cap F \neq \emptyset\}$. ◀

A minimização de D, $MIN(D)$, é um AFD mínimo equivalente a D. Este resultado é estabelecido na Proposição 1.67.

Exemplo 1.66 Considere-se o autómato D referido no Exemplo 1.42. Este autómato não é vacuoso e $UT(D) = D$. Recordando a partição $\{\{p\}, \{q\}, \{r, s\}\}$ induzida por Eqv_D calculada no Exemplo 1.61, e designando por C_0 o conjunto $\{p\}$, por C_1 o conjunto $\{q\}$ e por C_2 o conjunto $\{r, s\}$, conclui-se que a minimização de D é $MIN(D) = (Q_{\mathrm{m}}, \{a, b\}, \delta_{\mathrm{m}}, q_0^{\mathrm{m}}, F_{\mathrm{m}})$ em que

- $Q_{\mathrm{m}} = \{C_0, C_1, C_2\}$;
- $\delta_{\mathrm{m}} : Q_{\mathrm{m}} \times \{a, b\} \to Q_{\mathrm{m}}$ é tal que

δ	a	b
C_0	C_1	
C_1	C_1	C_2
C_2	C_1	C_2

- $q_0^{\mathrm{m}} = C_0$;
- $F_{\mathrm{m}} = \{C_1\}$. ▲

Enunciam-se de seguida as propriedades de $MIN(D)$ acima referidas.

Proposição 1.67 Dado um AFD D, $MIN(D)$ é um AFD mínimo equivalente a D. ■

Pode agora concluir-se que se um AFD não vacuoso não tem estados inúteis e não tem estados distintos equivalentes então o AFD é mínimo. Recorde-se que o caso dos AFDS vacuosos já foi referido anteriormente. Os AFDS mínimos para a linguagem vazia são os que têm um só estado e não têm transições nem estados finais.

Proposição 1.68 Se um AFD não vacuoso não tem estados inúteis e não tem estados distintos equivalentes então é um AFD mínimo.

Prova: Seja D um AFD não vacuoso sem estados inúteis e sem estados equivalentes distintos. Então $MIN(D)$ é igual a D, a menos do nome dos estados (que agora são vistos como conjuntos). Pela Proposição 1.67, D é AFD mínimo. ∎

Se dois AFDS são equivalentes e mínimos então eles são essencialmente o mesmo, a menos do nome dos respectivos estados, ou seja, são isomorfos.

Proposição 1.69 Dois AFDS equivalentes e mínimos são isomorfos. ∎