Capítulo 3

Minimização de AFD

Vimos que uma linguagem pode ser reconhecida por diferentes autómatos finitos. Mesmo quando nos restringimos à classe dos autómatos finitos deterministas constatamos que podem existir vários autómatos "distintos" para reconhecer a mesma linguagem. Somos assim confrontados com diversas questões.

A primeira delas é: quando é que dois AFD são "distintos"? Claramente dois autómatos que possuam um número diferente de estados devem ser considerados distintos (uma vez que requerem um número diferente de "componentes"). Podemos assim desde já pensar no problema de identificar e construir um autómato reconhecedor de uma determinada linguagem que possua o menor número de estados possível.

Mas será que existem dois autómatos mínimos "distintos" que reconhecem a mesma linguagem? E como é que se averigua se um autómato é ou não mínimo? E no caso de não ser mínimo como é que se constrói um autómato mínimo?

No que se segue, vamos debruçar-nos sobre as respostas a estas questões.

O problema de determinar um autómato mínimo equivalente a um dado autómato começou a ser estudado na década de 50 do século XX, com David Huffman (Estados Unidos da América, 1925 - 1999), em 1954, e Moore, em 1956. Um outro algoritmo mais eficiente, mas que vai para além dos objectivos deste livro, foi posteriormente desenvolvido por John Edward Hopcroft (Estados Unidos da América, 1939 -), em 1971.

Os algoritmos de minimização são geralmente aplicáveis apenas a autómatos deterministas. No entanto, em 1963, Janusz Brzozowski (Polónia, 1935 - 2019) desenvolveu um algoritmo de minimização que se aplica directamente a autómatos não deterministas [12].

O algoritmo que vamos apresentar inspira-se na publicação, de 1986, do trabalho de Alfred Aho (Canadá, 1941 -), Ravi Sethi (Índia, 1947 -) e Jeffrey Ullman (Estados Unidos da América, 1942 -) [6].

3.1 Distinguibilidade de estados

Na figura seguinte ilustramos dois autómatos finitos deterministas reconhecedores da linguagem $L = \{ba^n : \ n \in \mathbb{N}_0\}$.

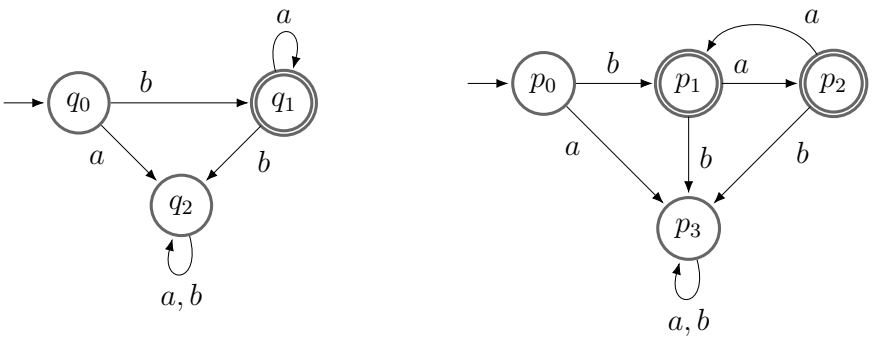


Figura 3.1.1 Dois autómatos reconhecedores da linguagem $L = \{ba^n : \ n \in \mathbb{N}_0\}$

Os dois estados p_1 e p_2 do autómato do lado direito da figura têm o "mesmo comportamento". De facto são ambos estados de aceitação, uma transição a partir deles pela letra a resulta num estado de aceitação e uma transição a partir deles pela letra b resulta no estado p_2.

Assim, as derivações de uma qualquer palavra nestes dois autómatos resultam ambas num estado de aceitação ou ambas num estado de rejeição, pelo que a sua funcionalidade é idêntica. Podemos assim "fundir" os dois estados p_1 e p_2 no autómato do lado direito, obtendo um autómato que é "idêntico" ao do lado esquerdo.

Definição 3.1.2 Linguagens Esquerda e Direita.

Seja $A = \big(Q, \Sigma, \delta, s, F\big)$ um autómato finito determinista.

A **linguagem esquerda** de um estado $q \in Q$ é

$$\mathcal{L}_{\text{esq}}(q) = \{w \in \Sigma^\star : \ \delta^\star(s, w) = q\}.$$

A **linguagem direita** de um estado $q \in Q$ é

$$\mathcal{L}_{\text{dir}}(q) = \{w \in \Sigma^\star : \ \delta^\star(q, w) \in F\}.$$

Se a linguagem esquerda de um estado é a linguagem vazia então o estado é inacessível a partir do estado inicial e pode ser removido. Para determinarmos todos os estados inacessíveis de um autómato basta enumerar todos os caminhos com origem no estado inicial e de comprimento inferior ao número total de estados do autómato. Os estados que não figurem em pelo menos um dos caminhos gerados serão inacessíveis.

Definição 3.1.3 Estado Acessível.

Um estado q de um autómato finito é **acessível** (a partir do estado inicial) sempre que $\mathcal{L}_{\text{esq}}(q) \neq \emptyset$. Caso contrário, é um **estado inacessível**.

Um *autómato* finito é *acessível* quando todos os seus estados são acessíveis.

A linguagem direita de um estado traduz o seu "comportamento". Assim, dois estados terão o mesmo "comportamento" se possuírem a mesma linguagem direita.

Definição 3.1.4 Estados Indistinguíveis/Distinguíveis.

Seja $A = (Q, \Sigma, \delta, s, F)$ um autómato finito determinista.

Dois estados $p, q \in Q$ são *indistinguíveis*, e escrevemos $p \equiv q$, sempre que $\mathcal{L}_{\text{dir}}(p) = \mathcal{L}_{\text{dir}}(q)$.

Dois estados $p, q \in Q$ são *distinguíveis*, e escrevemos $p \not\equiv q$, quando não são indistinguíveis, isto é, quando $\mathcal{L}_{\text{dir}}(p) \neq \mathcal{L}_{\text{dir}}(q)$.

Teorema 3.1.5

A relação de indistinguibilidade definida entre estados de um autómato finito determinista é uma relação de equivalência.

Demonstração.

Começamos por observar que a que a igualdade de linguagens corresponde à igualdade de conjuntos, a qual constitui uma relação de equivalência. Seja Q o conjunto de estados de um qualquer AFD.

Reflexiva. Para $q \in Q$, $q \equiv q$, uma vez que $\mathcal{L}_{\text{dir}}(q) = \mathcal{L}_{\text{dir}}(q)$.

Simétrica. Para $p, q \in Q$, $p \equiv q \implies q \equiv p$. De facto, se $\mathcal{L}_{\text{dir}}(p) = \mathcal{L}_{\text{dir}}(q)$ então também $\mathcal{L}_{\text{dir}}(q) = \mathcal{L}_{\text{dir}}(p)$.

Transitiva. Para $p, q, r \in Q$, $p \equiv q \wedge q \equiv r \implies p \equiv r$. De facto, se $\mathcal{L}_{\text{dir}}(p) = \mathcal{L}_{\text{dir}}(q)$ e se $\mathcal{L}_{\text{dir}}(q) = \mathcal{L}_{\text{dir}}(r)$ então também $\mathcal{L}_{\text{dir}}(p) = \mathcal{L}_{\text{dir}}(r)$.

Consideremos um qualquer AFD $A = \big(Q, \Sigma, \delta, s, F\big)$.

Dois estados são *indistinguíveis* sempre que se comportam do mesmo modo com *todas as palavras*:

$$\forall w \begin{cases} \text{se} \quad \delta^\star(p, w) \in F \text{ então } \delta^\star(q, w) \in F \\ \text{se} \quad \delta^\star(p, w) \notin F \text{ então } \delta^\star(q, w) \notin F \end{cases}.$$

Por outro lado, dois estados são *distinguíveis* quando se comportam de modo dife-

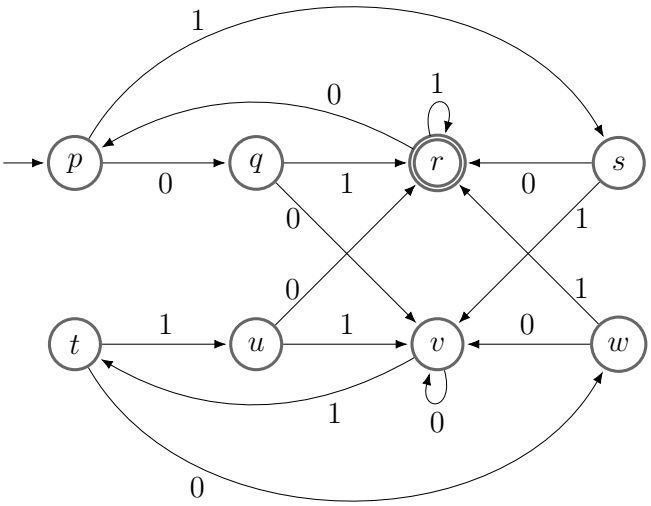
rente com *pelo menos uma palavra*:

$$\exists w, (\delta^\star(p,w) \in F \wedge \delta^\star(q,w) \notin F) \vee (\delta^\star(p,w) \notin F \wedge \delta^\star(q,w) \in F)\,.$$

Em particular, *qualquer estado de aceitação é distinguível de qualquer estado de rejeição.*

Exemplo 3.1.6

Vamos determinar alguns pares de estados distinguíveis e indistinguíveis do autómato seguinte.



Vejamos:

- $\{r, v\}$? Temos que $\delta^\star(r, \varepsilon) \in F$ e $\delta^\star(v, \varepsilon) \notin F$. Logo, $r \not\equiv v$.
- $\{s, w\}$? Temos que $\delta^\star(s, 0) = r \in F$ e $\delta^\star(w, 0) = v \notin F$. Logo, $s \not\equiv w$.
- $\{p, t\}$? Temos que:
  - $\delta^\star(p, \varepsilon) = p \notin F$ e $\delta^\star(t, \varepsilon) = t \notin F$. Logo, não distinguimos os estados pela palavra vazia.
  - $\delta^\star(p, 0) = q \notin F$ e $\delta^\star(t, 0) = w \notin F$. Logo, não distinguimos os estados pela palavra 0.
  - $\delta^\star(p, 1) = s \notin F$ e $\delta^\star(t, 1) = u \notin F$. Logo, não distinguimos os estados pela palavra 1.
  - $\delta^\star(p, 00) = v = \delta^\star(t, 00)$. Logo, $\delta^\star(p, 00x) = \delta^\star(v, x) = \delta^\star(t, 00x)$ para qualquer palavra $x \in \Sigma^\star$. Logo, não distinguimos os estados por palavras que começam com 00.

- $\delta^\star(p, 01) = r = \delta^\star(t, 01)$. Logo, $\delta^\star(p, 01x) = \delta^\star(r, x) = \delta^\star(t, 01x)$ para qualquer palavra $x \in \Sigma^\star$. Logo, não distinguimos os estados por palavras que começam com 01.
- $\delta^\star(p, 10) = r = \delta^\star(t, 10)$. Logo, $\delta^\star(p, 10x) = \delta^\star(r, x) = \delta^\star(t, 10x)$ para qualquer palavra $x \in \Sigma^\star$. Logo,não distinguimos os estados por palavras que começam com 10.
- $\delta^\star(p, 11) = v = \delta^\star(t, 11)$. Logo, $\delta^\star(p, 11x) = \delta^\star(v, x) = \delta^\star(t, 11x)$ para qualquer palavra $x \in \Sigma^\star$. Logo, não distinguimos os estados por palavras que começam com 11.

Dos pontos anteriores, concluímos que não distinguimos os estados p e t por qualquer palavra e portanto $p \equiv t$.

Mais à frente, após vermos uma forma de sistematizar a identificação de pares de estados distinguíveis, analisaremos os restantes pares de estados.

O algoritmo seguinte começa por marcar todos os pares de estados que são distinguíveis por uma palavra de tamanho 0, ou seja, pela palavra vazia.

Em seguida, iterativamente, utiliza os pares de estados que são distinguíveis por palavras de tamanho menor ou igual a k para marcar todos os pares de estados não marcados que sejam distinguíveis por uma palavra de tamanho $k + 1$.

O algoritmo termina após uma iteração em que não sejam marcados novos pares de estados.

Algoritmo 3.1.7 Identificação de Pares de Estados Distinguíveis.

Entrada: um AFD completo e acessível.

Saída: todos os pares de estados distinguíveis marcados.

Para cada $p \in F$

  Para cada $q \notin F$

    marcar o par $\{p, q\}$;

Repetir

  alterado $\leftarrow$ falso;

  Para cada par $\{p, q\}$ não marcado

marcou ← falso;

Para cada $a \in \Sigma$ enquanto não marcou

se o par $\{\delta(p,a), \delta(q,a)\}$ está marcado então

marcar o par $\{p, q\}$;

marcou ← verdadeiro;

alterado ← verdadeiro;

até não alterado.

Teorema 3.1.8

Se $\{p, q\}$ é um par de estados marcado pelo Algoritmo 3.1.7, p. 69 então $p \not\equiv q$.

Demonstração.

O funcionamento do algoritmo baseia-se no processo indutivo seguinte.

Caso base. Se $p \in F$ e $q \notin F$ então $p \not\equiv q$.

Passo indutivo. Se existe $a \in \Sigma$ tal que $\delta(p,a) \not\equiv \delta(q,a)$ então $p \not\equiv q$.

De facto, se existe $a \in \Sigma$ tal que $r = \delta(p,a)$ e $s = \delta(q,a)$ e sendo $r \not\equiv s$ então existirá uma palavra w que permite distinguir r de s. Logo, *ou* $\delta^\star(r,w) \in F$ *ou* $\delta^\star(s,w) \in F$. Assim, a palavra aw permite distinguir p de q uma vez que $\delta^\star(p,aw) = \delta^\star(r,w)$ e $\delta^\star(q,aw) = \delta^\star(s,w)$.

Os estados marcados no início do algoritmo correspondem ao *caso base* enquanto os marcados nas diversas iterações correspondem ao *passo indutivo*.

Teorema 3.1.9

Se $\{p, q\}$ é um par de estados não marcado pelo Algoritmo 3.1.7, p. 69 então $p \equiv q$.

Demonstração.

Assumamos que, depois de executar o algoritmo, existem pares de estados distinguíveis que não chegaram a ser marcados. A cada um desses pares associamos uma palavra de tamanho mínimo que distingue os dois estados do par.

Podemos assim fixar um par $\{p, q\}$ em que a palavra associada, w, tem tamanho mínimo entre todas as palavras associadas aos pares de estados em questão.

Atendendo a que todos os pares $\{\text{aceitação}, \text{rejeição}\}$ são marcados no início do

algoritmo, concluímos que $w \neq \varepsilon$ e portanto $w = ax$, para algum $a \in \Sigma$ e $x \in \Sigma^\star$.

Sejam $r = \delta(p, a)$ e $s = \delta(q, a)$. Uma vez que *ou* $\delta^\star(p, w) \in F$ *ou* $\delta^\star(q, w) \in F$, também *ou* $\delta^\star(r, x) \in F$ *ou* $\delta^\star(s, x) \in F$. Assim, o par $\{r, s\}$ é distinguível pela palavra x.

Se o par $\{r, s\}$ tivesse sido marcado pelo algoritmo numa determinada iteração então também o par $\{p, q\}$ teria sido marcado pelo algoritmo na iteração seguinte (e não foi!).

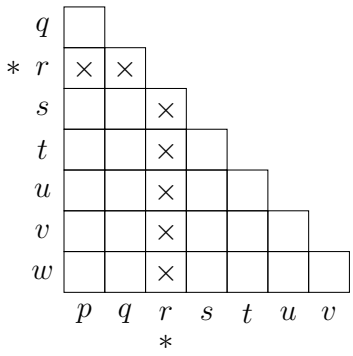
Se o par $\{r, s\}$ não tivesse sido marcado pelo algoritmo então seria um par não marcado distinguível pela palavra x de tamanho inferior ao tamanho da palavra w, o que é uma contradição com a escolha que fizemos de w.

Em qualquer dos casos chegamos a uma contradição, pelo que não podem existir pares de estados distinguíveis e não marcados após a execução algoritmo.

Exemplo 3.1.10

Ilustramos a aplicação do Algoritmo 3.1.7, p. 69 ao autómato representado no Exemplo 3.1.6, p. 68.

Inicialmente marcamos todos pares distinguíveis {aceitação, rejeição}:



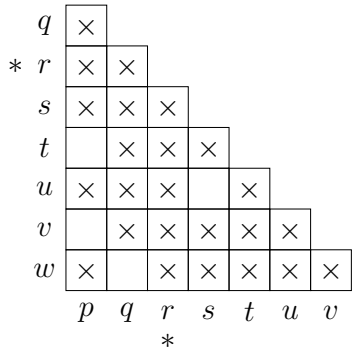
Em seguida, marcamos sucessivamente os restantes pares,analisando apenas as transições simples.

Iteração 1.

- $\{p, q\}$?
  - $\delta^\star(p, 0) = q \wedge \delta^\star(q, 0) = v$, mas $\{q, v\}$ ainda não é um par de estados distinguíveis. Logo, nada se conclui.
  - $\delta^\star(p, 1) = u \wedge \delta^\star(q, 1) = r$ e $\{u, r\}$ é um par de estados distinguíveis. Logo, o par $\{p, q\}$ também é distinguível.
- $\{p, s\}$?...

- ...

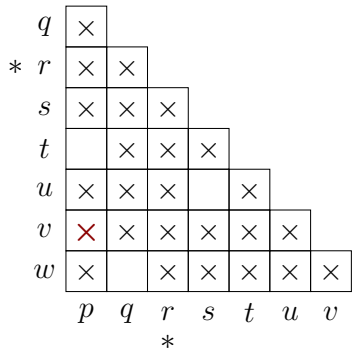
No final da 1ª iteração obtemos o quadro:



Iteração 2.

- $\{p, t\}$?
  - $\delta^\star(p, 0) = q \wedge \delta^\star(t, 0) = w$, mas $\{q, w\}$ ainda não é um par de estados distinguíveis. Logo, nada a concluir.
  - $\delta^\star(p, 1) = u \wedge \delta^\star(t, 1) = u$. Nada a concluir.
- $\{p, v\}$?
  - $\delta^\star(p, 0) = q \wedge \delta^\star(v, 0) = v$ e $\{q, v\}$ é um par de estados distinguíveis. Logo, o par $\{p, v\}$ também é distinguível.
- ...

No final da 2ª iteração obtém-se o quadro:



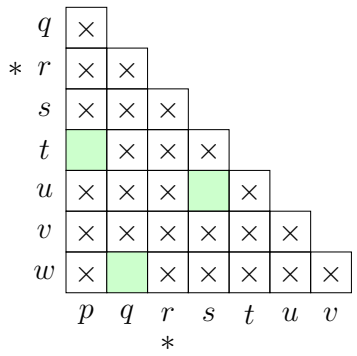
Iteração 3.

- $\{p, t\}$?

- $\delta^\star(p,0) = q \wedge \delta^\star(t,0) = w$, mas $\{q,w\}$ ainda não é um par de estados distinguíveis. Logo, nada a concluir.
- $\delta^\star(p,1) = u \wedge \delta^\star(t,1) = u$. Nada a concluir.

• ...

No final da 3ª iteração o quadro não sofre alterações e o algoritmo termina com o quadro:



Restam assim 3 pares de estados não marcados sendo estes os pares indistinguíveis:

$$p \equiv t, \quad q \equiv w, \quad s \equiv u.$$

3.2 O AFD mínimo

Na secção anterior desenvolvemos um método para identificar os estados indistinguíveis num autómato finito determinista.

A relação de indistinguibilidade de estados, $\equiv$, sendo uma relação de equivalência no conjunto de estados Q, induz uma partição em classes de equivalência $Q_\equiv = \{[q] : \ q \in Q\}$, onde $[q] = \{p \in Q : \ q \equiv p\}$ denota a classe de equivalência do estado $q \in Q$.

Cada uma destas classes de equivalência corresponde a um dos estados de um autómato equivalente ao original, chamado autómato reduzido. Os resultados apresentados ao longo desta secção mostram que o autómato reduzido é afinal o "menor" autómato equivalente ao autómato original, na medida em que não contém estados nem transições redundantes.

Definição 3.2.1 Autómato Quociente (Reduzido).

Seja $A = \big(Q, \Sigma, \delta, s, F\big)$ um autómato finito determinista e seja $\equiv$ a relação de indistinguibilidade de estados do autómato A.

O *autómato quociente* do autómato A pela relação $\equiv$ é o autómato finito determinista $A_{\equiv} = \big(Q_{\equiv}, \Sigma, \delta_{\equiv}, [s], F_{\equiv}\big)$, onde:

- o conjunto de estados $Q_{\equiv} = \{[q] : \ q \in Q\}$ é o conjunto das classes de equivalência de Q pela relação $\equiv$;
- o estado inicial $[s]$ é a classe de equivalência do estado inicial do autómato A;
- o conjunto de estados de aceitação $F_{\equiv} = \{[f] : \ f \in F\}$ é o conjunto de classes de equivalência de estados de aceitação do autómato A;
- a função de transição $\delta_{\equiv} : Q_{\equiv} \times \Sigma \to Q_{\equiv}$ é definida para $[q] \in Q_{\equiv}$ e $a \in \Sigma$ por
$$\delta_{\equiv}([q], a) = [\delta(q, a)].$$

A função de transição do autómato quociente está bem definida, uma vez que não depende do representante da classe escolhido. A justificação é dada pelo lema seguinte.

Lema 3.2.2

Seja $A = \big(Q, \Sigma, \delta, s, F\big)$ um autómato finito determinista e seja $\equiv$ a relação de indistinguibilidade de estados do autómato A.

Se $[p] = [q]$ então para $a \in \Sigma$, $[\delta(p, a)] = [\delta(q, a)]$.

Demonstração.

Para $w \in \Sigma^\star$, $\delta^\star\left(\delta(p, a), w\right) \in F$ se e só se $\delta^\star(p, aw) \in F$ se e só se $\delta^\star(q, aw) \in F$ se e só se $\delta^\star\left(\delta(q, a), w\right) \in F$.

Com o objectivo de verificar que o autómato quociente é equivalente ao autómato original, enunciamos dois resultados auxiliares.

Lema 3.2.3

Sejam Seja $A = \big(Q, \Sigma, \delta, s, F\big)$ um autómato finito determinista e $A_{\equiv}$ o autómato quociente pela relação de indistinguibilidade $\equiv$. Para qualquer palavra $w \in \Sigma^\star$ e estado $q \in Q$,
$$\delta^\star{}_{\equiv}([q], w) = \left[\delta^\star(q, w)\right].$$

Demonstração.
A prova faz-se por indução estrutural em $\Sigma^\star$, a qual deixamos ao cuidado do leitor, como exercício.

Lema 3.2.4

Se A é um autómato finito determinista e $A_\equiv$ é o autómato quociente pela relação de indistinguibilidade $\equiv$ então $f \in F$ se e só se $[f] \in F_\equiv$.

Demonstração.
A proposição é consequência imediata da definição de $F_\equiv$.

Vamos agora provar que um autómato determinista e o seu autómato quociente são equivalentes.

Teorema 3.2.5

Se A é um autómato finito determinista e $A_\equiv$ é o autómato quociente pela relação de indistinguibilidade $\equiv$ então $\mathcal{L}(A) = \mathcal{L}(A_\equiv)$.

Demonstração.
Uma qualquer palavra $w \in \mathcal{L}(A)$ se e só se $\delta^\star(s, w) \in F$ se e só se $[\delta^\star(s, w)] \in F_\equiv$ se e só se ${\delta^\star}_\equiv([s], w) \in F_\equiv$ se e só se $w \in \mathcal{L}(A_\equiv)$.

O autómato quociente pela relação de indistinguibilidade possui ainda a importante propriedade de não ser possível reduzi-lo mais.

Definição 3.2.6

Um AFD é *reduzido* se todos os seus estados são distinguíveis.

Teorema 3.2.7

O autómato quociente de um AFD *pela relação de indistinguibilidade é um autómato reduzido.*

Demonstração.
Os estados do autómato quociente de um AFD pela relação de indistinguibilidade são distinguíveis entre si, uma vez que correspondem às diferentes classes de equivalência pela relação de indistinguibilidade: a mesma palavra que serve para distinguir dois estados do AFD original pode ser utilizada para distinguir as duas classes de equivalência correspondentes a esses dois estados no autómato quociente. Deixamos ao cuidado do leitor, como exercício, desenvolver os detalhes da demonstração.

Exemplo 3.2.8

No Exemplo 3.1.10, p. 71 obtivemos 3 pares de estados indistinguíveis entre os 8 estados do autómato: $p \equiv t$, $q \equiv w$ e $s \equiv u$.

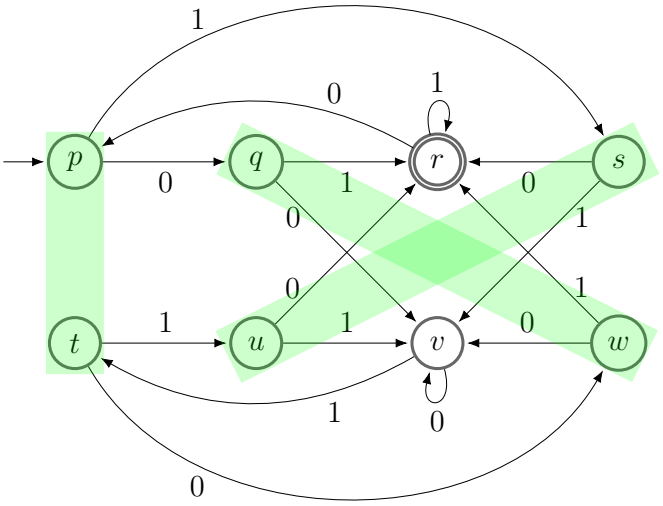


Figura 3.2.9 Autómato com 3 pares de estados indistinguíveis

Temos assim 5 classes de equivalência distintas que definem uma partição do conjunto de estados do autómato e que vão ser os estados do autómato quociente:

$$Q_{\equiv} = \{\{p,t\},\{q,w\},\{s,u\},\{r\},\{v\}\}.$$

Assim, a partir da tabela de transições associada ao autómato inicial

δ	0	1
$\rightarrow p$	q	s
q	v	r
$*r$	p	r
s	r	v

δ	0	1
t	w	u
u	r	v
v	v	t
w	v	r

construímos a tabela de transições associada ao autómato quociente utilizando as classes de equivalência calculadas

$\delta_{\equiv}$	0	1
$\rightarrow \{p,t\}$	$\{q,w\}$	$\{s,u\}$
$\{q,w\}$	$\{v\}$	$\{r\}$
$\{s,u\}$	$\{r\}$	$\{v\}$
$\{v\}$	$\{v\}$	$\{p,t\}$
$*\{r\}$	$\{p,t\}$	$\{r\}$

Esta última tabela corresponde ao diagrama de transições do autómato quociente a seguir ilustrado:

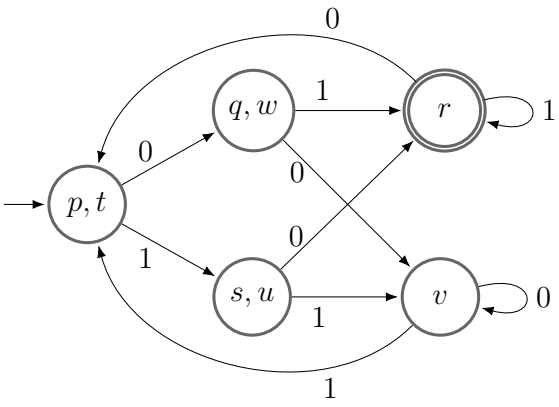


Figura 3.2.10 Autómato minimizado

Vamos agora mostrar que o AFD quociente é o menor autómato que reconhece a linguagem do autómato original e que, para além disso, é único.

Definição 3.2.11 Autómato Mínimo.

Um AFD é *mínimo* se não existir um autómato finito equivalente com um menor número estados.

Lema 3.2.12

Se um AFD *é mínimo então é acessível e reduzido.*

Demonstração.

Se um autómato possui estados inacessíveis então podemos eliminar esses estados e obter um autómato equivalente com menos estados. Logo, um autómato não acessível não pode ser mínimo.

Se um autómato possui um par de estados indistinguíveis então o autómato quociente terá menos estados. Logo, um autómato não reduzido não pode ser mínimo.

O autómato quociente de um AFD pela relação de indistinguibilidade é um autómato reduzido, uma vez que todos os seus estados (as classes de equivalência) são distinguíveis.

Exemplo 3.2.13

Os estados do autómato quociente obtido no fim do Exemplo 3.2.8, p. 76 são todos distinguíveis.

Por exemplo, vimos que os estados s e w do autómato inicial eram distinguíveis pela palavra 0, donde são também distinguíveis pela mesma palavra os estados $[s] = \{s, u\}$ e $[w] = \{q, w\}$ do autómato quociente.

Dizemos que dois autómatos são isomorfos quando têm exactamente a mesma "estrutura", ou seja, diferem apenas nos nomes dos estados. Temos assim a definição seguinte.

Definição 3.2.14

Sejam $A = \left(Q_A, \Sigma, \delta_A, s_A, F_A\right)$ e $B = \left(Q_B, \Sigma, \delta_B, s_B, F_B\right)$ dois AFD. Um *isomorfismo* de A em B é uma função bijectiva $h\colon Q_A \to Q_B$ que preserva

1. o estado inicial: $h(s_A) = s_B$;
2. os estados de aceitação: $h(F_A) = F_B$;
3. as transições: para $q \in Q_A$ e $a \in \Sigma$, $h(\delta_A(q, a)) = \delta_B(h(q), a)$.

Dois autómatos finitos deterministas são *isomorfos* se existir um isomorfismo entre eles.

É fácil verificar que a relação de isomorfismo na classe dos autómatos finitos deterministas é uma relação de equivalência. Em seguida, provamos que dois autómatos acessíveis e reduzidos que reconheçam a mesma linguagem são necessariamente isomorfos.

Lema 3.2.15

Dois quaisquer AFD *equivalentes acessíveis e reduzidos são isomorfos.*

Demonstração.

Sejam $A = \left(Q_A, \Sigma, \delta_A, s_A, F_A\right)$ e $B = \left(Q_B, \Sigma, \delta_B, s_B, F_B\right)$ AFD acessíveis e reduzidos. Uma vez que A é acessível, começamos por associar uma palavra w_q a cada estado $q \in Q_A$ que satisfaça $q = {\delta^\star}_A(s_A, w_q)$.

Definimos o isomorfismo $h\colon Q_A \to Q_B$ por $h(q) = {\delta^\star}_B(s_B, w_q)$, para $q \in Q_A$.

Para garantir que é um isomorfismo, precisamos de verificar que a função h:

a) está bem definida e não depende da escolha das palavras associadas aos estados;

b) é bijectiva;

c) satisfaz as propriedades da definição Definição 3.2.14, p. 78.

Vejamos:

a) Basta verificar que quaisquer que sejam as palavras z, w,

$$\delta^\star{}_A(s_A, z) = \delta^\star{}_A(s_A, w) \text{ se e só se } \delta^\star{}_B(s_B, z) = \delta^\star{}_B(s_B, w). \quad (3.2.1)$$

De facto, consideremos que $\delta^\star{}_A(s_A, z) = \delta^\star{}_A(s_A, w)$. Se fosse $\delta^\star{}_B(s_B, z) \neq \delta^\star{}_B(s_B, w)$ então, uma vez que o autómato B é reduzido, existiria uma palavra x que distinguiria o estado $\delta^\star{}_B(s_B, z)$ do estado $\delta^\star{}_B(s_B, w)$.

Nesse caso, ter-se-ia que *ou* $\delta^\star{}_B(s_B, zx) \in F_B$ *ou* $\delta^\star{}_B(s_B, wx) \in F_B$.

Consequentemente, apenas uma das duas palavras zx ou wx seria aceite pelo autómato B e, uma vez que os autómatos são equivalentes, apenas uma das duas palavras seria aceite pelo autómato A. Mas se $\delta^\star{}_A(s_A, z) = \delta^\star{}_A(s_A, w)$ então também $\delta^\star{}_A(s_A, zx) = \delta^\star{}_A(s_A, wx)$.

Assim, *ou* ambas as palavras seriam aceites *ou* nenhuma das palavras seria aceite.

Chegamos a uma contradição, pelo que quando $\delta^\star{}_A(s_A, z) = \delta^\star{}_A(s_A, w)$ então também $\delta^\star{}_B(s_B, z) = \delta^\star{}_B(s_B, w)$.

A implicação contrária prova-se de forma análoga, bastando trocar A com B.

b) Deixamos ao cuidado do leitor, como exercício, provar que h é bijectiva.

c) Quanto às propriedades da definição de isomorfismo, temos que:

1. Uma vez que ε é a única palavra que satisfaz $\delta^\star{}_A(s_A, \varepsilon) = s_A$, $h(s_A) = \delta^\star{}_B(s_B, \varepsilon) = s_B$.
2. É consequência dos dois autómatos reconhecerem a mesma linguagem que, para qualquer palavra w,

$$\delta^\star{}_A(s_A, w) \in F_A \text{ se e só se } \delta^\star{}_B(s_B, w) \in F_B. \quad (3.2.2)$$

Em particular, se w_f é a palavra associada a um estado $f \in F_A$ então $f = \delta^\star{}_A(s_A, w_f)$ e, portanto, $h(f) = \delta^\star{}_B(s_B, w_f) \in F_B$. Daqui concluímos que $h(F_A) \subseteq F_B$.

Por outro lado, atendendo a que h é sobrejectiva, a afirmação seguinte é verdadeira. Para qualquer estado de aceitação $f \in F_B$, existe $q \in Q_A$ tal que $h(q) = f$. Assim, se w_q é a palavra associada ao estado q então $q = \delta^\star{}_A(s_A, w_q)$ e $f = \delta^\star{}_B(s_B, w_q)$. A propriedade (3.2.2) implica que $q \in F_A$. Daqui concluímos que $F_B \subseteq h(F_A)$.

3. Sejam agora $q \in Q_A$ e $a \in \Sigma$. Se w_q é a palavra associada ao estado q então $h(q) = \delta^\star{}_B(s_B, w_q)$ e $\delta_B(h(q), a) = \delta^\star{}_B(s_B, w_q a)$.

   Por outro lado, $q = \delta^\star{}_A(s_A, w_q)$ e também $\delta_A(q, a) = \delta^\star{}_A(s_A, w_q a)$ pelo que $h(\delta_A(q, a))$ é igual a $\delta^\star{}_B(s_B, w_q a)$. Concluímos assim que $h(\delta_A(q, a)) = \delta_B(h(q), a)$.

Lema 3.2.16

Se um AFD *é acessível e reduzido então é mínimo.*

Demonstração.

Suponhamos que A é um AFD acessível e reduzido mas que não é mínimo. Então existe um AFD B, equivalente ao autómato A, com menos estados do que A e que é mínimo.

Pelo Lema 3.2.12, p. 77 o autómato B é também acessível e reduzido. Assim, A e B são dois AFD equivalentes, acessíveis e reduzidos e pelo Lema 3.2.15, p. 78 são isomorfos.

Porém, se A e B são isomorfos então têm o mesmo número de estados. Contradição! Assim, se A é um AFD acessível e reduzido então é necessariamente um AFD mínimo.

Teorema 3.2.17

Dois quaisquer autómatos mínimos equivalentes são isomorfos.

Demonstração.

Pelo Lema 3.2.12, p. 77, se dois autómatos equivalentes são mínimos então são também dois autómatos equivalentes acessíveis e reduzidos. Pelo Lema 3.2.15, p. 78, são ainda isomorfos.

Concluímos que qualquer autómato finito pode ser minimizado com a seguinte sequência de transformações:

1. se não é determinista, construímos um AFD usando o algoritmo de determinização;
2. se não é completo, construímos um AFD equivalente completo introduzindo um estado ratoeira;
3. se não é acessível, construímos um AFD equivalente, completo e acessível, eliminando os estados inacessíveis;
4. se não é reduzido, construímos um AFD equivalente, completo, acessível e reduzido, obtendo o autómato quociente pela relação de indistinguibilidade.

O AFD assim obtido é o autómato mínimo a menos de um isomorfismo.

Exemplo 3.2.18

Sendo completo, acessível e reduzido, o autómato quociente obtido no fim do Exemplo 3.2.8, p. 76 é o autómato mínimo para a linguagem reconhecida pelo autómato original (a menos de um isomorfismo).

3.3 Equivalência entre AFD

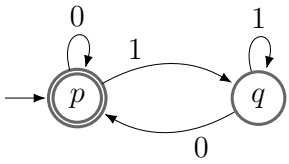
O método de identificação de estados distinguíveis que estudámos na Secção 3.1, p. 66 permite também construir um teste para equivalência de autómatos finitos.

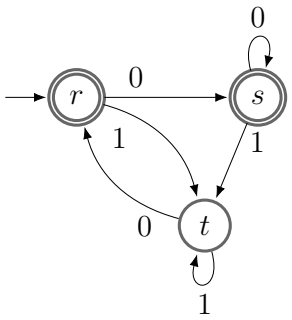
Algoritmo 3.3.1 Teste de Equivalência de AFD.

1. Juntar os dois autómatos na mesma tabela (assumindo que os conjuntos de estados são disjuntos);
2. Usando o Algoritmo 3.1.7, p. 69 para identificação de pares distinguíveis, *os dois autómatos são equivalentes* se e só se *os seus estados iniciais são indistinguíveis.*

Exemplo 3.3.2

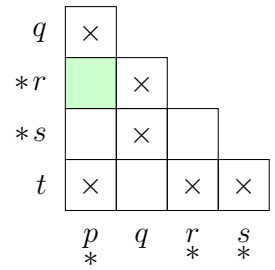
Consideremos os seguintes AFD:





Serão Equivalentes?

Ambos os autómatos são acessíveis e os conjuntos de estados são disjuntos. Assim, juntando-os numa única tabela e aplicando o algoritmo de identificação de estados distinguíveis, obtemos:



Uma vez que os dois estados iniciais são indistinguíveis, isto é, $p \equiv r$, concluímos que os dois AFD são equivalentes: ambos reconhecem a linguagem

$$L = \{w \in \{0,1\}^\star : \ w \text{ é a palavra vazia ou termina em } 0\}.$$

3.4 Exercícios

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

2. Demonstre o Lema 3.2.3, p. 74.

3. Mostre que a função h referida na demonstração do Lema 3.2.15, p. 78 é bijectiva

4. Demonstre o Teorema 3.2.7, p. 75.

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

(a)

δ	0	1
$\rightarrow p$	q	r
$*q$	p	r
$*r$	p	q

(b)

δ	0	1
$\rightarrow *p$	p	q
q	q	r
$*r$	r	q

(c)

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

(d)

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

(e)

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

(f)

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

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

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

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

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.

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

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.

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

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

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 $L = (L_1 \cup L_2)^\star$.

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

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

10. Aplique o Algoritmo 3.3.1, p. 81 para testar a equivalência dos autómatos ilustrados na Figura 3.1.1, p. 66.