Capítulo 3

Autómatos Finitos Não Deterministas (NFA)

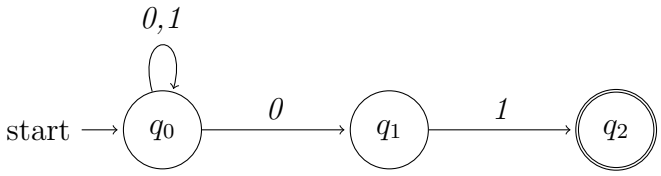
Os autómatos finitos não deterministas (ou NFAs, do inglês *Non-Deterministic Finite Automata*) são uma extensão dos DFAs, permitindo que um determinado símbolo de entrada possa conduzir a mais do que um estado. Este não-determinismo não acrescenta novas capacidades a estes autómatos, em comparação com os DFAs, mas pode permitir simplificar muito a criação de autómatos para resolução de alguns problemas (no entanto, esta 'simplificação' pode acarretar uma maior complexidade na interpretação e no desenho dos NFAs). Vamos também ver como os NFAs podem ser transformados em DFAs equivalentes.

Definição

A definição de um NFA continua a ser o tuplo $NFA = (Q, \Sigma, \delta, q_0, F)$, mas em que agora δ representa a função de transição de estados e símbolos para um conjunto de estados ($\delta(q, a) = M$, em que $M \subseteq Q$ representa o conjunto dos estados possíveis de destino.

Este não-determinismo leva a que seja necessário manter em aberto todas as possibilidades de transição do NFA. Para que uma cadeia seja aceite pelo NFA é necessário que pelo menos um dos 'caminhos' leve a um estado final. Isto pode ser visto como um processamento em paralelo de todas as possibilidades.

Vamos ver o exemplo de um NFA que permita reconhecer cadeias no alfabeto $\{0, 1\}$ terminadas em 01. Vejamos a solução em forma de diagrama:



Com esta solução, e para processar uma cadeia, '*escolhemos*' permanecer em q_0 até ao penúltimo símbolo de entrada. Se este for 0, leva-nos a q_1, e se o último símbolo for 1, o NFA fica assim em q_2, aceitando a cadeia.

Por exemplo, para a cadeia '1001', ao processar o primeiro 1, o NFA permanece em q_0. Ao ler o segundo símbolo (0), o NFA poderia nessa altura transitar para q_1, ou permanecer em q_0, pelo que temos agora de manter as duas hipóteses em aberto. Ao receber o terceiro símbolo (0), caso o NFA esteja em q_1 irá morrer, pois não há nenhuma transição definida; caso esteja em q_0, temos novamente duas transições possíveis: q_0 e q_1. Ao receber o último símbolo (1), temos novamente duas hipóteses: se o NFA estiver em q_0, transita novamente para q_0; se estiver em q_1, transita para q_2. Assim, no final da cadeia '1001', o NFA está ou em q_0 ou em q_2; como pelo menos um destes estados é um estado final, podemos dizer que a cadeia 1001 é aceite por este NFA.

Por outro lado, para a cadeia '0110', ao processar o primeiro 0, o NFA pode permanecer em q_0 ou ir para q_1. Ao receber o segundo símbolo (1), se estiver em q_0 mantém-se em q_0, e se estiver em q_1 vai para q_2. Ao receber o terceiro símbolo (1), se estiver em q_0, mantém-se em q_0, e se estiver em q_2 morre, pois não há nenhuma transição possível. Ao receber o último símbolo (0), e como apenas q_0 se mantém em aberto, podemos transitar para q_0 ou q_1, o que significa que a cadeia não é aceite, pois nenhum destes estados é final.

As tabelas abaixo mostram estas possibilidades de transição de uma forma mais compacta.

1	0	0	1
q_0	q_0	q_0	q_0
		q_1	q_2
	q_1	-	

0	1	1	0
q_0	q_0	q_0	q_0
			q_1
q_1	q_2	-	

As outras representações possíveis para o NFA (formal e tabular) mantêm-se também, com a diferença de as transições serem feitas para um conjunto de estados. Vamos ver então as representações formal e tabular para o NFA acima:

$$NFA = (\{q_0, q_1, q_2\}, \{0, 1\}, \{\delta(q_0, 0) = \{q_0, q_1\}; \delta(q_0, 1) = \{q_0\};$$
$$\delta(q_1, 0) = \emptyset; \delta(q_1, 1) = \{q_2\}; \delta(q_2, 0) = \emptyset; \delta(q_2, 1) = \emptyset\}, q_0, \{q_2\})$$

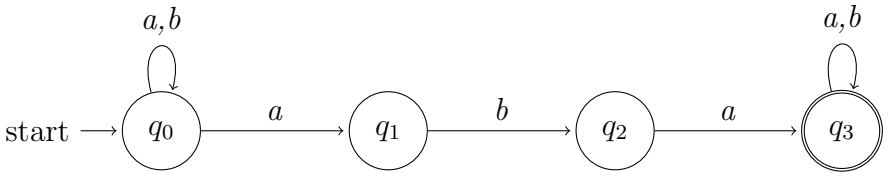
	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
$* q_2$	$\emptyset$	$\emptyset$

Repare-se que quando não existem transições, é usado o símbolo de conjunto vazio, $\emptyset$, para expressar esse facto.

Exercício 1

Obtenha um NFA que permita reconhecer cadeias no alfabeto {a,b} que contenham a sub-cadeia *aba*.

Para resolver este exercício, podemos ver que a única coisa que interessa é saber se a sequência 'aba' ocorre na cadeia de entrada. Então, podemos modelar o seguinte NFA:



Se olharmos para a solução, aquilo que temos nas transições entre q_0 e q_3 é o reconhecimento da cadeia 'aba'. Em q_0 a transição para o próprio estado permite que o NFA permaneça nesse estado até 'à altura certa', isto é, até que a sub-cadeia *aba* apareça na entrada. Após reconhecimento da cadeia *aba*, chegando ao estado q_3, ficamos em q_3, que é um estado de aceitação.

Olhando para o exemplo da cadeia abbabab, temos as seguintes possibilidades de transição, representadas na tabela abaixo.

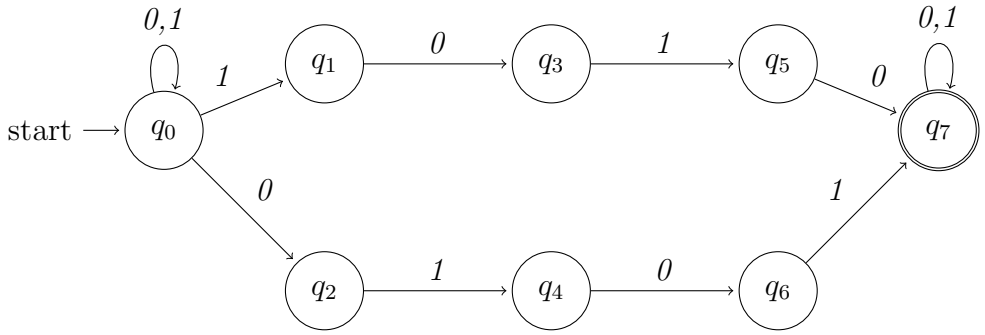
Como podemos ver, existe um caminho que leva a q_3 quando a sequência *aba* aparece na entrada, e permanece em q_3 até ao final do processamento da cadeia, o que permite aceitar a cadeia abbabab.

<table>
<tr><td>a</td><td>b</td><td>b</td><td>a</td><td>b</td><td>a</td><td>b</td></tr>
<tr><td rowspan="3">q_0</td><td rowspan="3">q_0</td><td rowspan="3">q_0</td><td rowspan="2">q_0</td><td rowspan="2">q_0</td><td>q_0</td><td>q_0</td></tr>
<tr><td>q_1</td><td>q_2</td></tr>
<tr><td>q_1</td><td>q_2</td><td>q_3</td><td>q_3</td></tr>
<tr><td>q_1</td><td>q_2</td><td>-</td><td></td><td></td><td></td><td></td></tr>
</table>

Exercício 2

Modele um NFA (tirando o máximo partido do não determinismo) que reconheça cadeias do alfabeto binário que contenham a sequência 1010 ou a sequência 0101.

Neste exercício temos algo semelhante ao anterior, mas em que queremos reconhecer uma coisa ou outra. Podemos modelar isto com uma divisão do caminho possível em dois 'ramos':



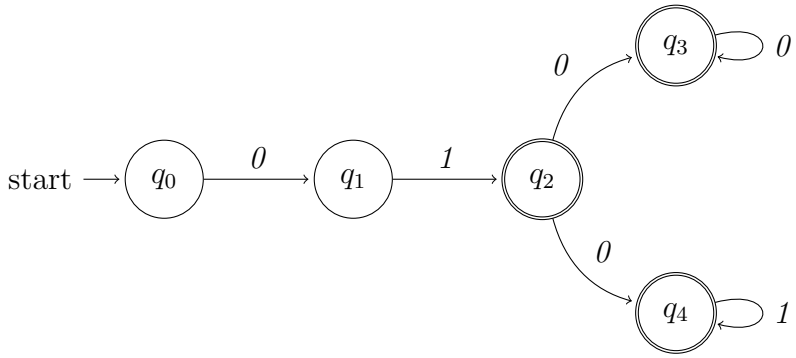
Podemos ver que esta solução é muito semelhante à solução para o problema anterior, mas desta vez com dois caminhos alternativos entre o estado inicial e o estado final.

Exercício 3

Obtenha um NFA com no máximo 5 estados para a linguagem das cadeias binárias no formato 0101^n ou 010^n ($n \geq 0$).

Neste exercício temos não só o desafio de modelar a linguagem como um NFA mas também o limite de 5 estados.

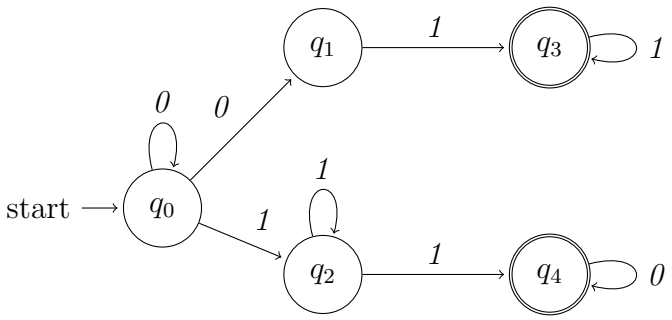
Se não tivessemos esse limite, poderíamos separar logo no início o NFA em dois ramos e cada um trataria da sua sub-linguagem. Assim, com esta limitação, podemos tentar aproveitar o facto de ambas as possibilidades de cadeias da linguagem começarem por 01 e colocar essa parte do caminho conjuntamente, e separar apenas quando houver necessidade disso.



Como podemos ver na solução, temos a parte inicial do NFA em comum para o reconhecimento do início das cadeias (01). Depois, temos a divisão em dois ramos, sendo que o ramo de cima permite reconhecer as cadeias com pelo menos um zero, o que juntando ao 01 já reconhecidos permite reconhecer as cadeias do formato 010^n com $n \geq 1$. Para permitir reconhecer com $n = 0$, o estado q_2 também necessita de ser final. Depois, no ramo de baixo, temos a possibilidade de reconhecer cadeias no formato 01^n, com $n \geq 0$, o que juntando ao 01 iniciais permite as cadeias no formato 0101^n com $n \geq 0$. Não poderíamos continuar a ter os estados q_3 e q_4 juntos, como até então, porque de um lado temos obrigatoriamente um zero, e do outro lado temos zero ou mais zeros, e não conseguiríamos distinguir os dois casos se a transição com 0 a partir de q_2 fosse para o mesmo estado.

Exercício 4

Considere o NFA abaixo.



Indique $\delta\hat{}\,(q_0, S)$ para cada uma das cadeias C: 00, 11, 1100, 1101, 011, 0011. Indique quais destas cadeias são aceites pelo NFA. Descreva a linguagem aceite pelo NFA.

Para determinar $\delta\hat{}\,(q_0, S)$ temos de verificar a cada passo quais os possíveis estados atingíveis, para no final termos o conjunto total de estados atingíveis. Se virmos no formato tabela como acima, equivale a indicar todos os estados presentes na última coluna da tabela. Assim, temos:
$\delta\hat{}\,(q_0, 00) = \{q_0, q_1\}$
$\delta\hat{}\,(q_0, 11) = \{q_2, q_4\}$
$\delta\hat{}\,(q_0, 1100) = \{q_4\}$
$\delta\hat{}\,(q_0, 1101) = \emptyset$
$\delta\hat{}\,(q_0, 011) = \{q_3, q_2, q_4\}$
$\delta\hat{}\,(q_0, 0011) = \{q_3, q_2, q_4\}$

As cadeias aceites são as que incluem pelo menos um estado de aceitação do NFA, ou seja, 11, 1100, 011 e 0011.

Em relação à linguagem aceite pelo NFA, temos de pensar nos caminhos que levam a estados de aceitação, neste caso os ramos de cima e de baixo. Olhando para o ramo de cima, temos um qualquer número de 0s aceite no estado inicial, seguido de um 0 na transição de q_0 para q_1 seguido de um 1 na transição para q_3, podendo aqui aceitar um qualquer número de 1s. Assim, temos um conjunto de cadeias no formato 0^i1^j com $i \geq 1, j \geq 1$.

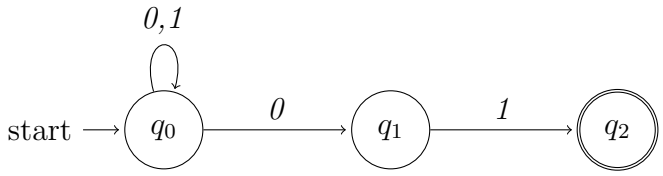
Olhando para o ramo de baixo, temos novamente um qualquer número de 0s no estado inicial, seguido de um 1 na transição para q_2, o qual por sua vez aceita um qualquer número de 1s. Temos novamente um 1 obrigatório na transição para q_4, o qual permite depois um qualquer número de 0s. Assim, temos um conjunto de cadeias no formato $0^i1^j0^k$ com $i \geq 0, j \geq 2, k \geq 0$.

Juntando os dois, obtemos então uma definição da linguagem do DFA: $\{0^i1^j : i \geq 1, j \geq 1\} \cup \{0^i1^j0^k : i \geq 0, j \geq 2, k \geq 0\}$.

Conversão de NFA para DFA

A conversão de NFA para DFA pode-se fazer facilmente através da técnica de construção de sub-conjuntos. Esta técnica simula o paralelismo dos caminhos possíveis do NFA, ao construir um DFA cujos estados representam as combinações de estados do NFA original a cada momento.

Voltando ao exemplo do NFA para reconhecer cadeias no alfabeto $\{0, 1\}$ terminadas em 01, recordemos o NFA:



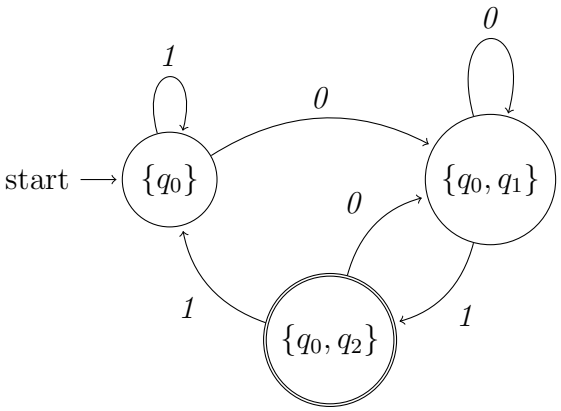
Para passar para DFA, podemos construir a tabela do DFA, começando com a primeira linha, que será o estado inicial do NFA:

	0	1
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$

Continuamos agora a preencher a tabela, adicionando na coluna de estados os que ainda não se encontram presentes, e nas colunas das transições, colocam-se todos os estados de destino possíveis. Como podemos ver na tabela acima, o estado constutuído por q_0 já se encontra na tabela, e portanto precisamos apenas de acrescentar o estado constituído pelos estados q_0 e q_1. Como podemos ver na tabela abaixo, já completa, a partir deste estado que contém q_0 e q_1 podemos transitar com 0 para esse mesmo estado, e com 1 para um novo estado que contém q_0 e q_2, o qual necessita de ser acrescentado à tabela.

	0	1
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$^* \{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$

No final, quando já todos os estados de destino de transições estiverem presentes na tabela, chegamos ao final da conversão. Neste caso, obtivemos na mesma 3 estados (embora em teoria o número de estados do DFA possa atingir 2^N, em que N é o número de estados do NFA, na prática é raro ultrapassar em muito o número de estados do NFA). Vejamos o DFA resultante graficamente:

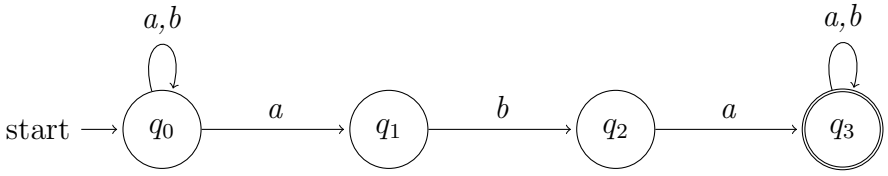


Apesar de manter o mesmo número de estados (3), o DFA apresenta mais transições do que o NFA equivalente.

Exercício 5

Converta o NFA obtido no Exercício 1 para o DFA equivalente.

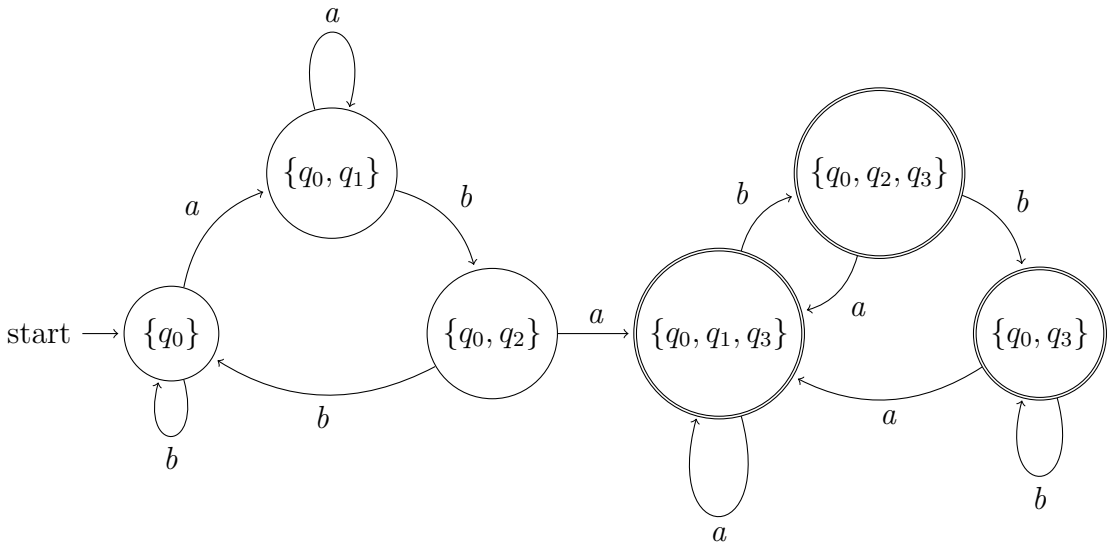
Recordando o DFA do exercício 1:



Convertendo para DFA, temos então a seguinte tabela:

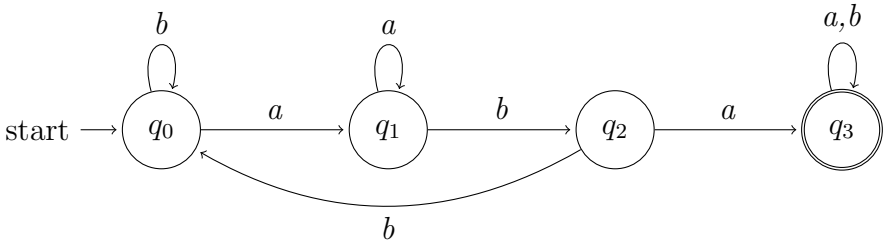
	a	b
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$\{q_0, q_2\}$	$\{q_0, q_1, q_3\}$	$\{q_0\}$
* $\{q_0, q_1, q_3\}$	$\{q_0, q_1, q_3\}$	$\{q_0, q_2, q_3\}$
* $\{q_0, q_2, q_3\}$	$\{q_0, q_1, q_3\}$	$\{q_0, q_3\}$
* $\{q_0, q_3\}$	$\{q_0, q_1, q_3\}$	$\{q_0, q_3\}$

Representando o DFA graficamente:



Podemos ver que neste caso obtemos mais estados do que no NFA original.

Podemos também tentar comparar com uma solução obtida diretamente ao pensar num DFA para resolver o problema. Ao tentar reconhecer a sequência aba, necessitavamos apenas de estados que mantivessem informação sobre que parte da sequência foi já reconhecida. Uma solução típica seria:

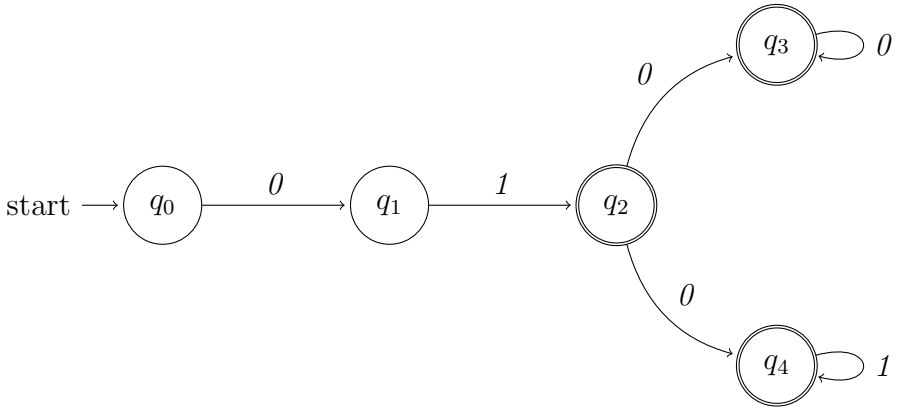


Mais à frente vamos falar sobre como minimizar o número de estados de um DFA, para eliminar estados desnecessários. Nessa altura, podemos voltar a este exemplo, para mostrar que os 3 estados finais obtidos acima pela conversão do NFA são equivalentes entre si, e portanto conseguimos minimizar esse DFA para um igual ao que construímos diretamente.

Exercício 6

Converta o NFA obtido no Exercício 3 para o DFA equivalente.

Relembrando o NFA do exercício 3:



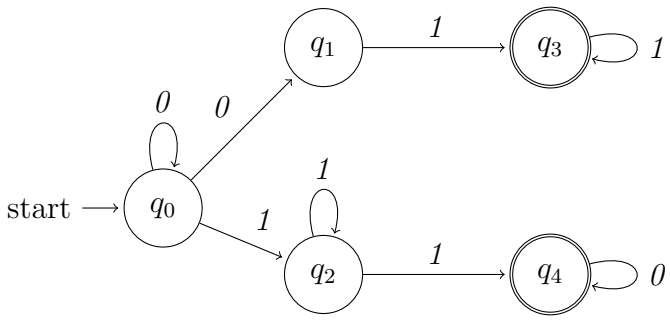
Ora, contruíndo então a tabela do DFA equivalente:

	0	1
$\rightarrow \{q_0\}$	$\{q_1\}$	$\emptyset$
$\{q_1\}$	$\emptyset$	$\{q_2\}$
$* \{q_2\}$	$\{q_3, q_4\}$	$\emptyset$
$* \{q_3, q_4\}$	$\{q_3\}$	$\{q_4\}$
$* \{q_3\}$	$\{q_3\}$	$\emptyset$
$* \{q_4\}$	$\emptyset$	$\{q_4\}$

Exercício 7

Converta o NFA do Exercício 4 para o DFA equivalente.

Relembrando o NFA do exercício 4:



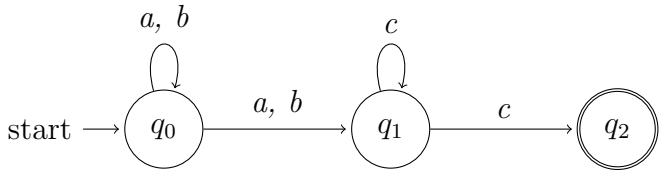
	0	1
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_2\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_2, q_3\}$
$\{q_2\}$	$\emptyset$	$\{q_2, q_4\}$
$^* \{q_2, q_3\}$	$\emptyset$	$\{q_2, q_3, q_4\}$
$^* \{q_2, q_4\}$	$\{q_4\}$	$\{q_2, q_4\}$
$^* \{q_2, q_3, q_4\}$	$\{q_4\}$	$\{q_2, q_3, q_4\}$
$^* \{q_4\}$	$\{q_4\}$	$\emptyset$

Construíndo a tabela do DFA equivalente:

Como podemos ver, o DFA tem 7 estados, em comparação com os 5 estados do NFA, o que fica ainda bastante aquém dos 32 (2^5) estados possíveis. Este é então mais um exemplo em que o número de estados do DFA não é muito superior ao número de estados do NFA que lhe deu origem.

Exercício 8 (TPC 2010/11)

Considere o autómato finito abaixo sobe o alfabeto $\Sigma = \{a, b, c\}$. Represente o autómato usando a notação formal e converta-o para um DFA que aceite a mesma linguagem, desenhando o DFA completo resultante.



Olhando rapidamente para o autómato, podemos ver que ele reconhece a linguagem das cadeias com pelo menos um a ou um b, e com pelo menos um c, e em que todos os c's aparecem no final da cadeia.

Para a representação na notação formal, temos apenas de lembrar a representação formal como um tuplo com 5 elementos e representar a informação nesse formato, ficando então com:

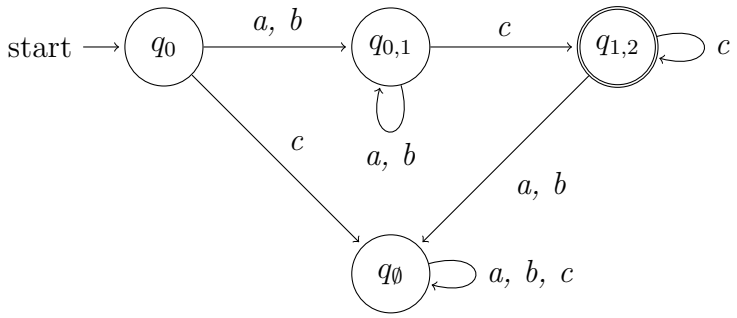
$(Q, \Sigma, q_0, \delta, F)$

em que $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{a, b, c\}$, $F = \{q_2\}$ e δ é definido por $\delta(q_0, a) = \{q_0, q_1\}$; $\delta(q_0, b) = \{q_0, q_1\}$; $\delta(q_0, c) = \emptyset$; $\delta(q_1, a) = \emptyset$; $\delta(q_1, b) = \emptyset$; $\delta(q_1, c) = \{q_1, q_2\}$; $\delta(q_2, a) = \emptyset$; $\delta(q_2, b) = \emptyset$; $\delta(q_2, c) = \emptyset$

Transformando este NFA num DFA, começamos novamente por q_0 e construímos a tabela de transições

	a	b	c
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\emptyset$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_1, q_2\}$
$* \{q_1, q_2\}$	$\emptyset$	$\emptyset$	$\{q_1, q_2\}$
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

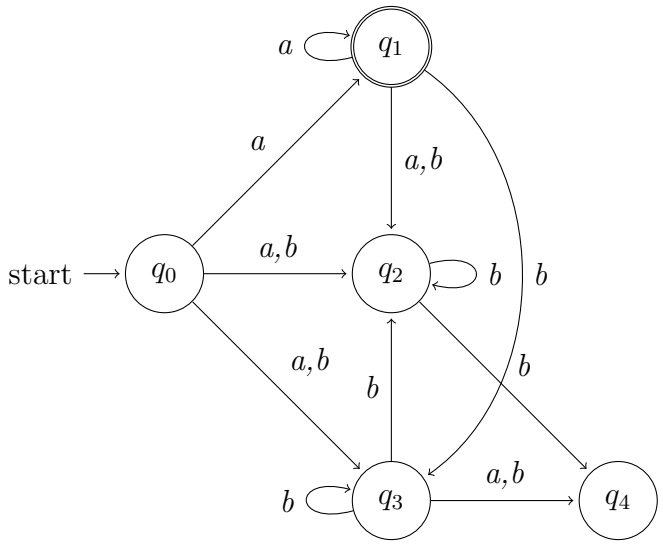
A partir da tabela podemos então desenhar também o DFA completo.



Como podemos ver, o DFA não tem muitos mais estados do que o NFA equivalente, sendo apenas acrescentado o estado morto para completar o DFA.

Exercício 9 (TPC 2011/12)

Represente o DFA abaixo usando a notação formal e converta-o para um DFA equivalente, apresentando o diagrama resultante.



Para representar o DFA na notação formal, temos apenas de representar os vários componentes:

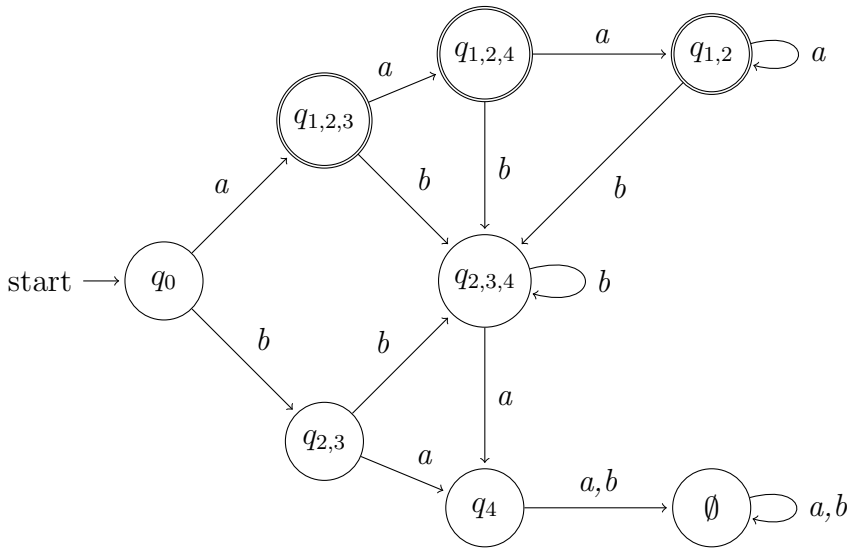
(Q, Σ, δ, q_0, F) = ($\{q_0, q_1, q_2, q_3, q_4\}$, {a, b}, δ, q_0, $\{q_1\}$)

em que δ é definido por: $\delta(q_0, a) = \{q_1, q_2, q_3\}$; $\delta(q_0, b) = \{q_2, q_3\}$; $\delta(q_1, a) = \{q_1, q_2\}$; $\delta(q_1, b) = \{q_2, q_3\}$; $\delta(q_2, b) = \{q_2, q_4\}$; $\delta(q_3, a) = \{q_4\}$; $\delta(q_3, b) = \{q_2, q_3, q_4\}$.

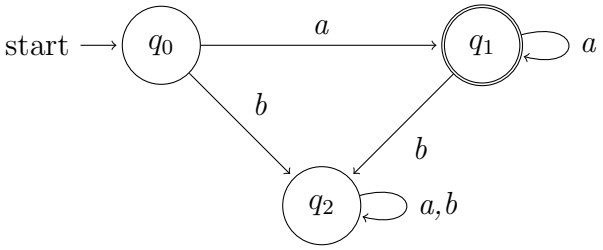
Para obter o DFA equivalente, podemos então usar o método de construção de sub-conjuntos, obtendo a seguinte tabela:

	a	b
$\rightarrow \{q_0\}$	$\{q_1, q_2, q_3\}$	$\{q_2, q_3\}$
* $\{q_1, q_2, q_3\}$	$\{q_1, q_2, q_4\}$	$\{q_2, q_3, q_4\}$
$\{q_2, q_3\}$	$\{q_4\}$	$\{q_2, q_3, q_4\}$
* $\{q_1, q_2, q_4\}$	$\{q_1, q_2\}$	$\{q_2, q_3, q_4\}$
$\{q_2, q_3, q_4\}$	$\{q_4\}$	$\{q_2, q_3, q_4\}$
$\{q_4\}$	$\emptyset$	$\emptyset$
* $\{q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2, q_3, q_4\}$
$\emptyset$	$\emptyset$	$\emptyset$

Podemos agora desenhar o DFA obtido:



Repare-se que o DFA não é muito maior do que o NFA original (8 estados vs 5 estados no NFA original). Note-se ainda que este DFA não é o DFA ideal para o reconhecimento desta linguagem. Na verdade, olhando para o NFA original, podemos ver que a linguagem reconhecida pelo autómato é a das cadeias constituídas apenas por a's e de comprimento maior ou igual a 1 (os estados q_2, q_3 e q_4 não permitem conduzir a um estado de aceitação). Para reconhecer esta mesma linguagem, bastava-nos ter um simples DFA com dois estados (mais o estado morto):



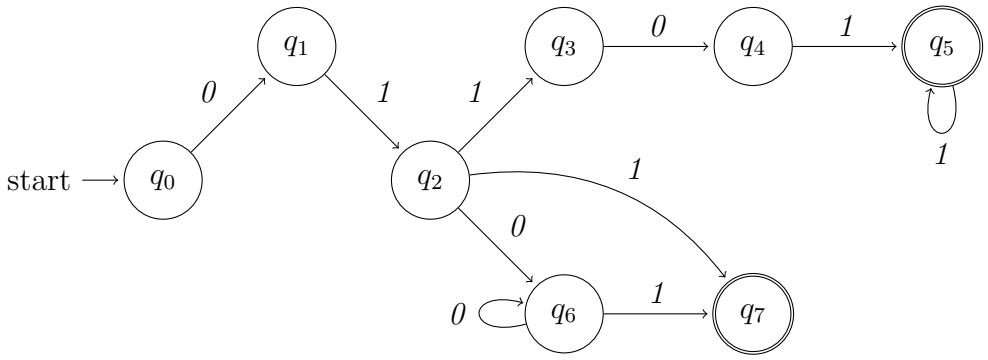
Mais tarde vamos então abordar a minimização de DFAs, e o método que nos permite chegar do primeiro DFA ao segundo.

Exercício 10 (Desafio 2014/15)

Modele um NFA que permita reconhecer cadeias do alfabeto {0,1} no formato $01101^i, i \geq 1$ ou no formato $010^j1, j \geq 0$.

Ex: 01101 pertence à linguagem; 0110 não pertence à linguagem; 011 pertence à linguagem; 0101 pertence à linguagem.
Converta o NFA obtido para o DFA equivalente.

Olhando para a linguagem, podemos simplesmente separar o autómato em dois ramos, e tratar cada um deles independentemente. No entanto, como ambos os ramos começam com os mesmos símbolos, podemos juntar a parte inicial, e assim reduzir o número de estados do NFA. Uma possível solução seria então:

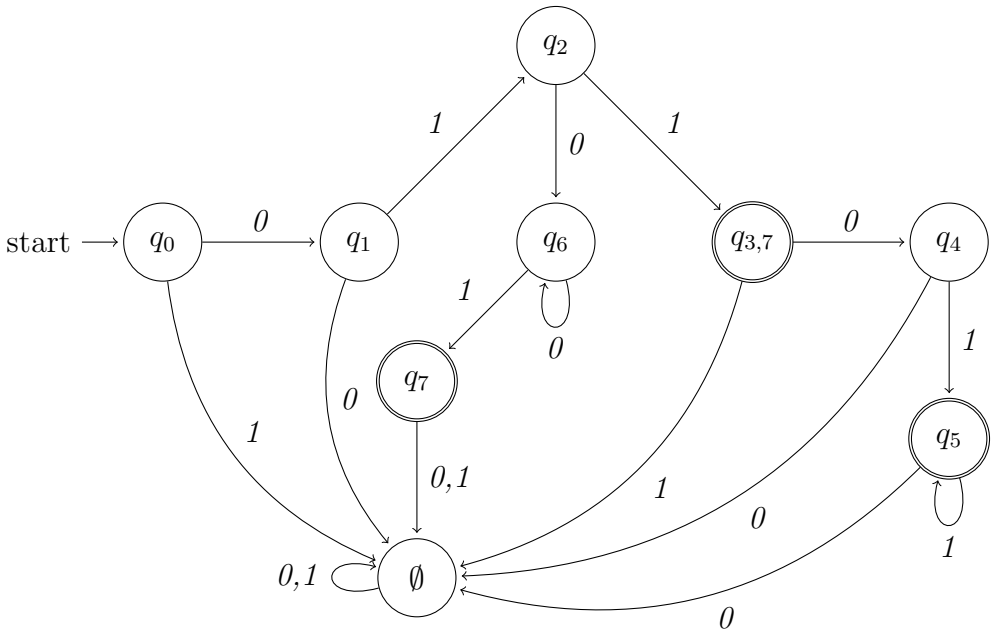


Olhando para a solução, vemos então a parte inicial e coincidente dos dois ramos nas primeiras transições, e a partir de q_2 os dois ramos para as duas partes da linguagem. A parte de cima reflete a primeira parte $(01101^i, i \geq 1)$, e a parte de baixo para a segunda parte $(010^j1, j \geq 0)$.

Convertendo para DFA, ficamos com a seguinte tabela:

	0	1
$\rightarrow \{q_0\}$	$\{q_1\}$	$\emptyset$
$\{q_1\}$	$\emptyset$	$\{q_2\}$
$\{q_2\}$	$\{q_6\}$	$\{q_3, q_7\}$
$\{q_6\}$	$\{q_6\}$	$\{q_7\}$
* $\{q_3, q_7\}$	$\{q_4\}$	$\emptyset$
* $\{q_7\}$	$\emptyset$	$\emptyset$
$\{q_4\}$	$\emptyset$	$\{q_5\}$
* $\{q_5\}$	$\emptyset$	$\{q_5\}$
$\emptyset$	$\emptyset$	$\emptyset$

Desenhando o DFA ficamos então com:



Novamente, podemos constatar que o número de estados do DFA resultante não é muito maior do que o número de estados do NFA original.

Exercício Proposto 1 Desenhe um NFA que aceite a linguagem das cadeias binárias em que o último dígito é diferente do primeiro. Converta o NFA obtido para um DFA e compare-o com a solução do Exercício 4 do capítulo anterior.

Exercício Proposto 2 Desenhe um NFA que aceite a linguagem das cadeias binárias com um número ímpar de zeros e número par de uns. Converta o NFA obtido para um DFA e compare os dois.