Capítulo 10

Máquinas de Turing

As Máquinas de Turing (TM, do inglês *Turing Machine*) são autómatos capazes de reconhecer ou processar qualquer tipo de linguagem. Podem ser vistas como uma máquina com uma cabeça de processamento que a cada momento se encontra numa posição de uma fita unidimensional infinita, e que se pode deslocar para a esquerda ou para a direita, mantendo ou alterando o símbolo presente na fita.

Formalmente uma TM é constituída por um conjunto Q de estados, um conjunto Σ de símbolos do alfabeto, um conjunto Γ de símbolos da fita, as transições δ associadas, o estado inicial $q_0 \in Q$, o símbolo (pertencente a Γ) que representa a inexistência de informação na fita (Branco) e o conjunto de estados finais $F \subseteq Q$. Uma transição é dependente do estado em que a máquina se encontra e qual o símbolo que está presente na posição atual da fita, indicando qual o estado de destino, o símbolo que passará a substituir o símbolo atual na fita, e a direção de movimentação da cabeça (esquerda ou direita). Formalmente, TM $= (Q, \Sigma, \Gamma, \delta, q_0, B, F)$, sendo as transições definidas como $\delta(q, X) = (p, Y, D)$, em que $q, p \in Q$, $X, Y \in \Gamma$, e D indica a direção de movimento (esquerda ou direita).

Uma TM pode ser representada por um autómato em que em cada transição se indica o símbolo atual na fita, o símbolo pelo qual é substituído e a direção de movimento da cabeça. De forma a conseguir acompanhar a evolução da computação numa TM, pode-se também apresentar a sequência de descrições instantâneas, sendo que em cada descrição instantânea se apresenta o conteúdo da fita e o estado atual da cabeça, sendo este representado imediatamente à esquerda do símbolo atual, de forma a transmitir também a sua posição.

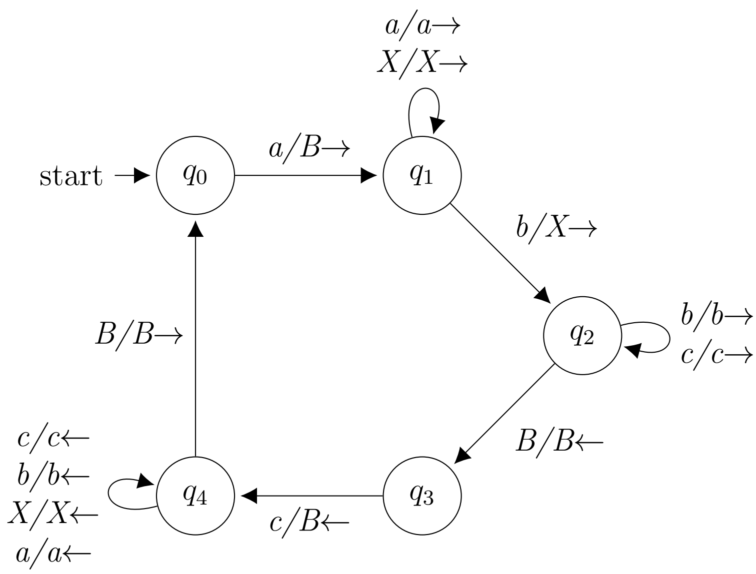
Vejamos um exemplo de uma TM para reconhecimento de cadeias no formato $a^n b^n c^n, n \geq 1$, que sabemos não ser uma CFL e por isso impossível de modelar usando autómatos de pilha ou gramáticas livres de contexto.

O primeiro passo para construir uma TM passa por delinear um algoritmo que permita resolver o problema tendo em conta o funcionamento de uma TM. Uma possível solução consiste em fazer várias passagens pela cadeia de forma a garantir que o número de a's, b's e c's é igual, marcando as partes da cadeia já processadas de forma a evitar contar duas vezes a mesma entrada. Como não existe preocupação em garantir que no final do processamento a cadeia se mantenha inalterada, podemos optar por apagar 'as pontas' da cadeia (os a's e os c's) e marcar apenas os b's já processados. Olhando para a cadeia $aaabbbccc$ como exemplo, a ideia será obter uma TM que permita os seguintes passos, assumindo o símbolo X como marca para os b's já processados:

$$BaaabbbcccB$$
$$BBaaXbbccBB$$
$$BBBaXXbcBBB$$
$$BBBBXXXBBBB$$

Neste ponto final, não havendo mais a's ou c's e se não houver mais b's (isto é, todos estarão transformados em X's), então a cadeia será aceite.

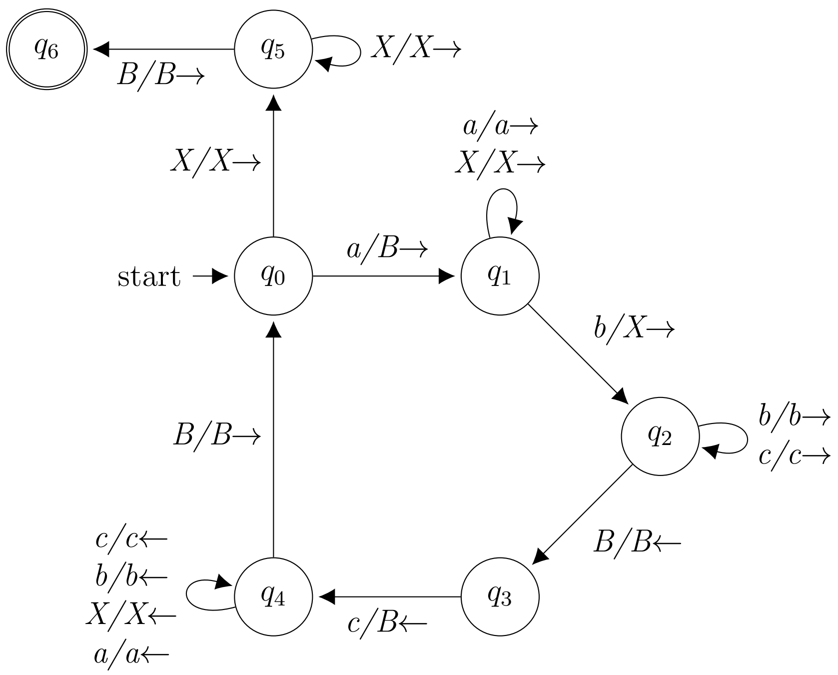
Tendo um algoritmo que funcione para a resolução do problema, podemos passar para o segundo passo, que consiste na implementação do algoritmo usando uma TM. Uma das 'partes' da máquina terá de lidar com as passagens iniciais pela cadeia, apagando o primeiro a, marcando o primeiro b que encontrar, e apagando o último c. Isso pode ser realizado pela seguinte TM parcial:



Analisando esta parte da TM, podemos ver que, partindo de q_0, o primeiro a é reconhecido e apagado (substituído por B). Depois, em q_1, a máquina anda para a direita 'vendo' a's ou X's até encontrar o primeiro b, o qual substitui por X, passando para q_2. Uma vez em q_2, a máquina continua para a direita através de b's e c's até ao final da cadeia, altura em que, passando para q_3, apaga o último c, passando para q_4. Em q_4 voltamos atrás, até atingir o início da cadeia, altura em que volta a q_0.

Falta-nos agora garantir que caso não seja encontrado mais nenhum a, todos os b's foram convertidos em X's, e todos os c's foram apagados.

Para isso, temos de acrescentar essa possibilidade, acrescentando uma nova possibilidade em q_0 de encontrar logo um X e não um a. Uma possibilidade de solução seria então:



Embora na primeira fase não seja validado que o formato é o correto (por exemplo, a cadeia *aaabbbcbccc* poderia ser processada), este último passo completa as garantias relativas ao formato e tamanho da cadeia, passando ao estado de aceitação (q_6) apenas nos casos devidos. Quando a cadeia não está no formato requerido, haverá um momento em que não há mais transições possíveis. Por exemplo, se a cadeia não terminar em c, em q_3 não haverá transição possível, e a TM morre nesse ponto; não se tratando de um estado de aceitação, a cadeia não é aceite.

A Máquina de Turing pode ainda ser representada de forma tabular, colocando num eixo os estados e no outro os símbolos do alfabeto da fita, sendo as células preenchidas com as transições respetivas. Neste exemplo, teríamos a seguinte tabela:

	a	b	c	X	B
$\rightarrow q_0$	(q_1, B, R)			(q_5, X, R)	
q_1	(q_1, a, R)	(q_2, X, R)		(q_1, X, R)	
q_2		(q_2, b, R)	(q_2, c, R)		(q_3, B, L)
q_3			(q_4, B, L)		
q_4	(q_4, a, L)	(q_4, b, L)	(q_4, c, L)	(q_4, X, L)	(q_0, B, R)
q_5				(q_5, X, R)	(q_6, B, R)
$*q_6$					

Os estados estão indicados nas linhas, enquanto os símbolos da fita estão presentes nas colunas. Podemos ver que cada célula contém o destino da transição correspondente. Foi utilizado L e R para indicar movimento da cabeça para a esquerda e direita, respetivamente. Uma célula em branco significa que não existe transição definida, sendo que a máquina termina a sua execução se se encontrar numa configuração que não permita qualquer transição.

Vamos ainda ver a sequência de descrições instantâneas para o processamento das cadeias abc e $aabcbc$:

$Bq_0abcB \vdash BBq_1bcB \vdash BXq_2cB \vdash BXcq_2B \vdash BXq_3cB \vdash Bq_4XBB \vdash Bq_4BXB \vdash Bq_0XB \vdash BXq_5B \vdash BXBq_6B$

Neste momento, não havendo mais nenhuma transição possível, o processamento termina. Dado que q_6 é um estado de aceitação, a cadeia abc é aceite pela TM.

Note-se que a indicação do símbolo B é opcional, podendo ser omitida exceto quando se encontra imediatamente à direita da cabeça da máquina.

$q_0aabcbc \vdash q_1abcbc \vdash aq_1bcbc \vdash aXq_2cbc \vdash aXcq_2bc \vdash aXcbq_2c \vdash aXcbcq_2B \vdash aXcbq_3c \vdash aXcq_4b \vdash aXq_4cb \vdash aq_4Xcb \vdash q_4aXcb \vdash q_4BaXcb \vdash q_0aXcb \vdash q_1Xcb \vdash Xq_1cb$

Neste momento, não existe transição definida, pelo que o processamento termina. Como q_1 não é um estado de aceitação, esta cadeia não é aceite.

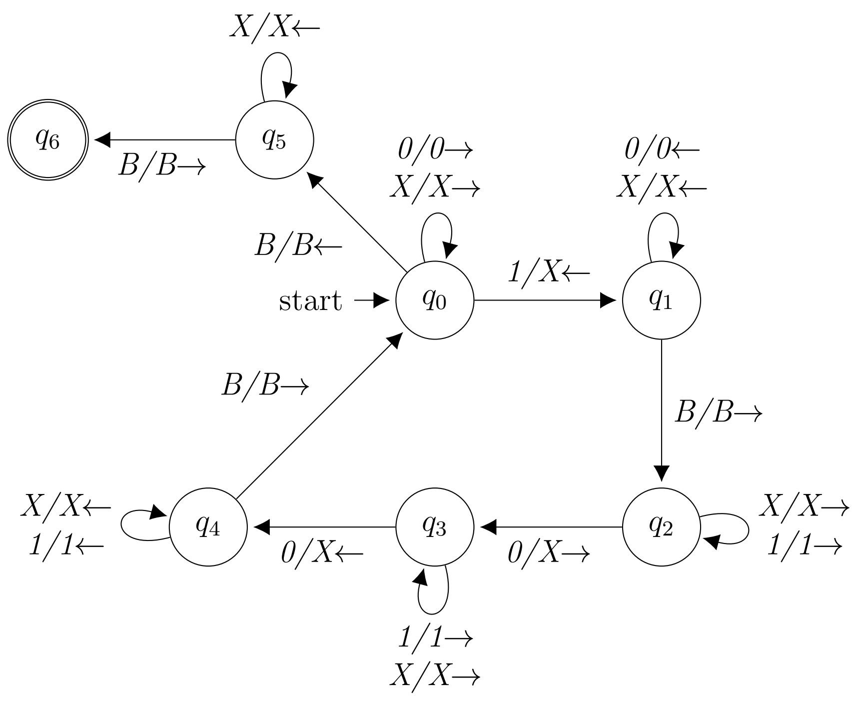
Exercício 1

Construa uma TM que permita reconhecer cadeias pertencentes à lingugem das cadeias binárias em que o número de zeros é o dobro do número de uns.

Novamente, para obtenção de uma solução começamos por delinear um algoritmo que permita resolver o problema, tratando depois da implementação. Como possível algoritmo para solução deste problema, podemos por exemplo realizar várias passagens pela cadeia, em que em cada passagem é feita a deteção de um 1, marcando-o como processado, e, voltando ao início, detetar dois 0's, marcando-os também. Novamente, será necessário no final garantir que caso não existam mais 1's também não existirão mais 0's.

Embora este algoritmo não seja muito eficiente, dado que regressa sempre ao início da cadeia, e obriga à deteção dos 1s e 0s na mesma ordem, torna-se mais simples de implementar.

Passando então para a implementação do algoritmo, podemos ter a seguinte TM como possível solução:

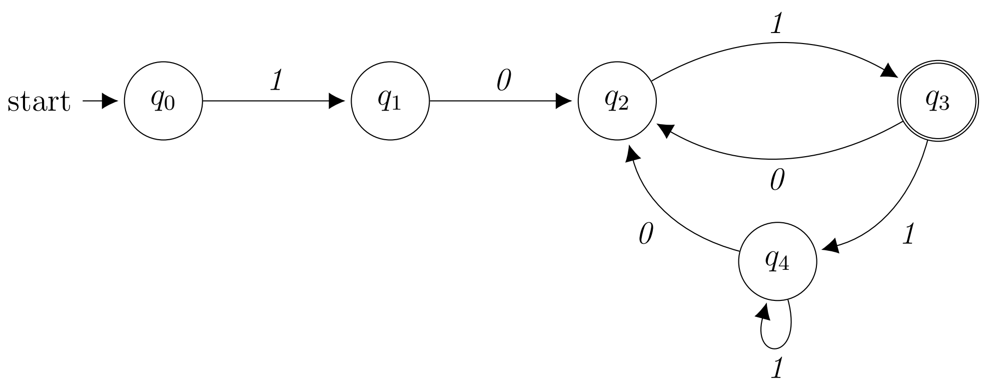


Podemos ver que começamos em q_0 por procurar o primeiro 1, avançando através dos possíveis 0's e X's existentes. Depois de identificar o 1, em q_1 volta-se ao início da cadeia, passando em q_2 a procurar o primeiro 0 não processado, e em q_3 o segundo 0, sendo que em q_4 voltamos novamente ao início da cadeia, para prosseguir para a próxima iteração. No caso de não existência de mais 1's, passa-se de q_0 para q_5, onde se verifica que não sobraram 0's, passando a q_6 apenas nesse caso.

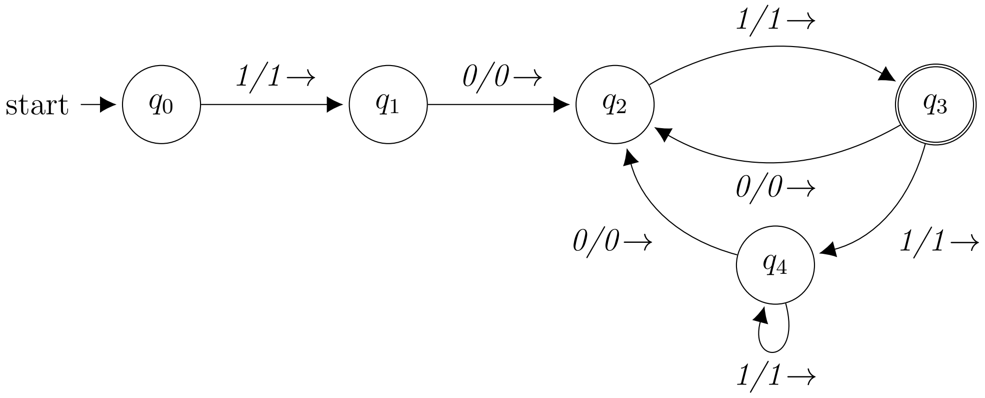
Exercício 2

Apresente uma Máquina de Turing que permita reconhecer cadeias binárias que iniciem em 10, terminem em 01, e não possuam nenhuma sequência com dois 0's.

A linguagem pedida neste exercício é uma Linguagem Regular, e como tal é possível definir um DFA para a reconhecer. De facto, este problema foi já visto no **Exercício 3** do segundo capítulo, em que foi produzido um DFA, que se recorda aqui (já sem o estado q_5, que não permite levar à aceitação):



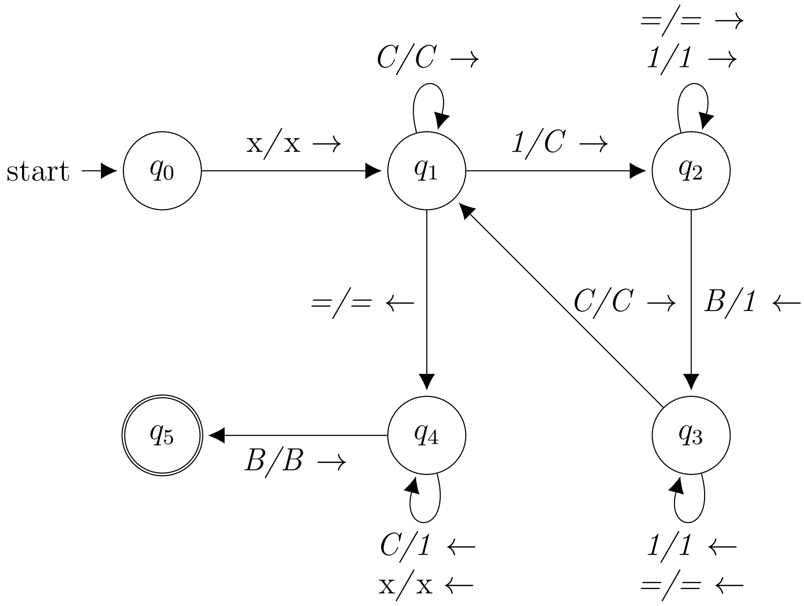
Uma TM para linguagens regulares pode ser vista como um DFA que mantém a cadeia na fita ao invés de a consumir. Então, modificando apenas as etiquetas das transições do diagrama do DFA, é possível obter o diagrama de uma TM para reconhecimento da mesma linguagem que o DFA original. Podemos então ficar com a seguinte TM:



Uma vez que a TM vai parar apenas quando não houver mais movimentos possíveis, isso acontecerá quando aparecer um 'Branco', sendo a cadeia aceite caso esteja em q_3 nesse momento, e não sendo aceite caso contrário.

Exercício 3 (Exame 2014/15)

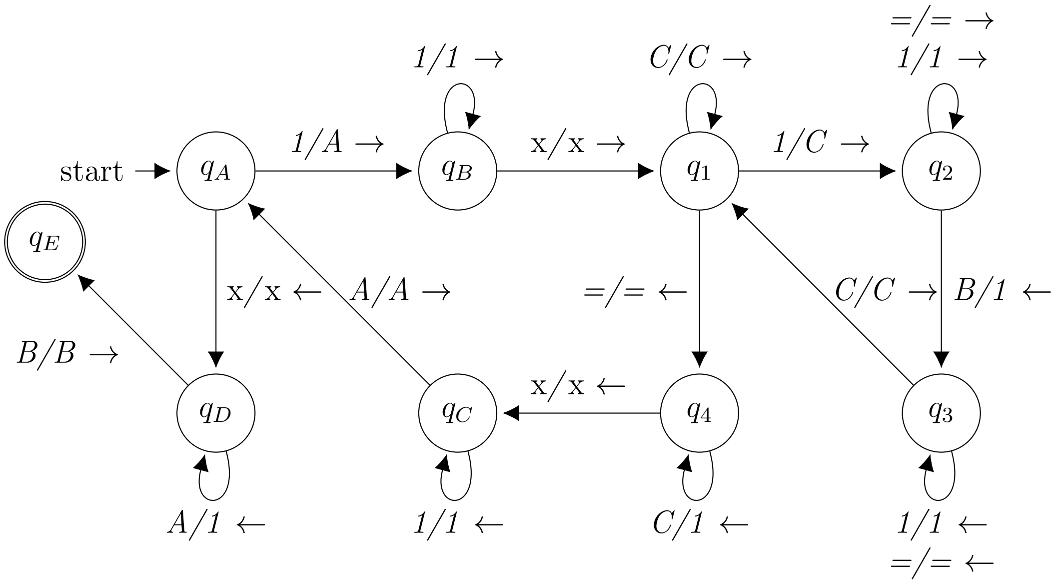
Pretende-se uma Máquina de Turing (TM) que realize a multiplicação de dois números em unário (em unário, os números são representados pela quantidade de 1's existentes; por exemplo, o número 3 em unário é representado por 111). A TM abaixo apresenta uma parte da solução.
a) Indique o que faz a TM apresentada e exemplifique, mostrando a sequência de descrições instantâneas da máquina quando a entrada na fita é x11=.
b) Baseando-se na TM apresentada, desenhe uma Máquina de Turing que implemente a multiplicação de dois números em unário. A entrada na fita é no formato N1xN2=, devendo no final a fita ficar com N1xN2=N3 em que N3 é a multiplicação de N1 por N2 em unário. Exemplo: 111x11= deverá resultar em 111x11=111111.



Começando então por analisar a máquina apresentada, podemos ver que ela inicia por reconhecer o x (interpretado aqui como sendo o sinal de multiplicação). É depois executado um ciclo em que são lidos C's em q_1 e o primeiro 1 é substituído por um C, passando em q_2 o resto da entrada até ao final (símbolos 1 e =), altura em que é inserido um novo 1 na cadeia, sendo que em q_3 se regressa até ao ponto em que o último 1 foi substituído por um C, voltando a q_1 e repetindo este ciclo até que não hajam mais 1's antes do sinal de igualdade. Nesta altura, passamos para q_4, percorrendo a cadeia para a esquerda e substituíndo os C's novamente por 1's.

Abstraíndo este funcionamento para uma linguagem de mais alto-nível, podemos dizer que esta TM copia os 1's entre o x e o = para depois do =. Vendo então o exemplo da cadeia x11=, o resultado será x11=11, como podemos ver pela sequência de descrições:
q_0x11= $\vdash$ x$q_1$11= $\vdash$ x$Cq_2$1= $\vdash$ xC1q_2= $\vdash$ xC1=q_2B $\vdash$ xC1q_3=1 $\vdash$ x$Cq_3$1=1 $\vdash$ xq_3C1=1 $\vdash$ x$Cq_1$1=1 $\vdash$ xCCq_2=1 $\vdash$ xCC=$q_2$1 $\vdash$ xCC=1q_2B $\vdash$ xCC=$q_3$11 $\vdash$ xCCq_3=11 $\vdash$ xCq_3C=11 $\vdash$ xCCq_1=11 $\vdash$ xCq_4C=11 $\vdash$ xq_4C1=11 $\vdash$ q_4x11=11 $\vdash$ q_4Bx11=11 $\vdash$ q_5x11=11

Em relação à TM completa, e de forma a reaproveitar a TM apresentada, podemos pensar em utilizá-la como uma rotina a invocar por cada 1 presente na parte à esquerda do sinal de multiplicação, ou seja, repetindo o segundo operando tantas vezes quantas o número de 1's no primeiro operando.



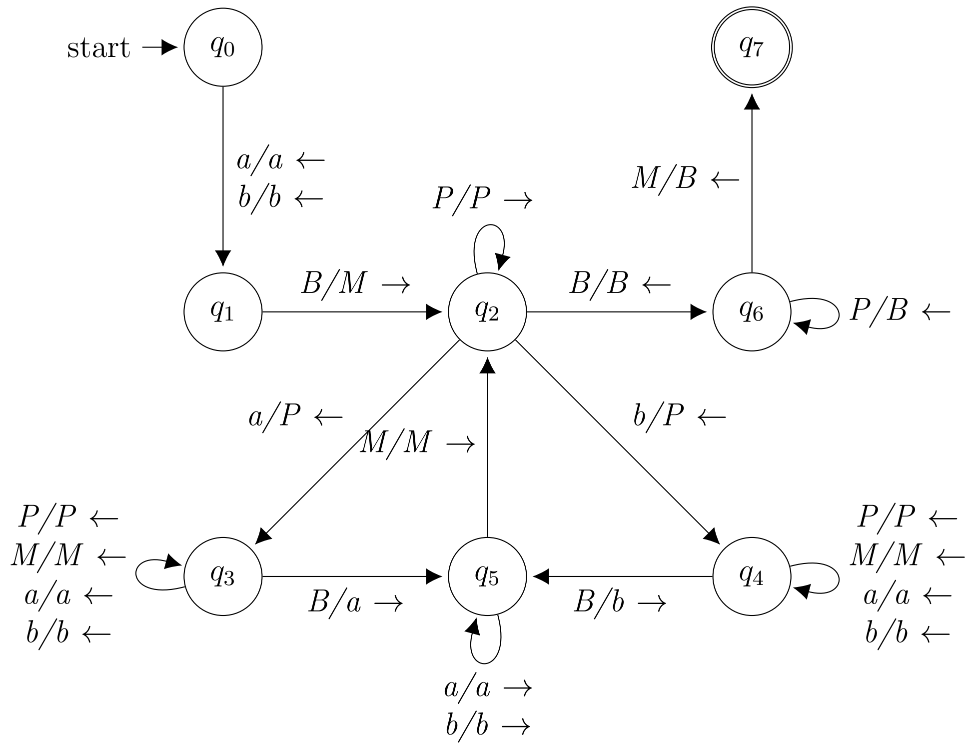
Repare-se que os estados q_0 e q_5 foram substituídos por q_B e q_C, respetivamente, com alterações ligeiras nas suas ligações, dada a necessidade de fazer a ponte entre a sub-rotina e a parte restante da TM, e para que a primeira seja reutilizável no contexto da TM geral.

Podemos ver entre q_A, q_B e q_C o mesmo padrão de ciclo, sendo 'invocada' a sub-TM em cada ciclo, ou seja, uma vez por cada 1 presente à esquerda do x. Finalmente, quando termina este ciclo, e é detetado um x em q_A passa-se para q_D, onde é feita a substituição dos A's de volta para 1's, deixando no final a cadeia inalterada.

Exercício 4 (TPC 2010/11)

Considere uma máquina de Turing que inverta uma sequência de símbolos no alfabeto Σ={a,b} existente na fita. Ex: BbabaB → BababB (B representa células vazias).
a) Desenhe a Máquina de Turing correspondente.
b) Apresente o traço de computação quando a entrada na fita é *bba*.

Para inverter a sequência na fita, podemos ir copiando cada um dos símbolos para a esquerda da cadeia original, em ordem inversa, apagando no final a cadeia original. Uma TM que poderá fazer isso poderá ser a seguinte:



Olhando para a TM acima, podemos ver que começa por andar para a esquerda, inserindo (em q_1) um marcador (símbolo M) para separar a parte original da cadeia da sua réplica invertida. Voltando depois para a direita, em q_2 verifica-se qual o próximo símbolo não processado (a ou b), marcando-o como processado (símbolo P) e transitando para q_3 ou q_4 consoante o símbolo lido. Em ambos estes estados, percorre-se a cadeia para a esquerda até encontrar uma célula vazia, inserindo nesse local o símbolo lido anteriormente e transitando para q_5. Percorre-se novamente a cadeia para a direita até chegar ao marcador de separação das cadeias, transitando novamente para q_2, e repetindo o ciclo até que não hajam mais símbolos por processar. Nessa altura, transita-se para q_6, percorrendo a cadeia original e apagando os símbolos P e M, terminando em q_7 em caso de sucesso.

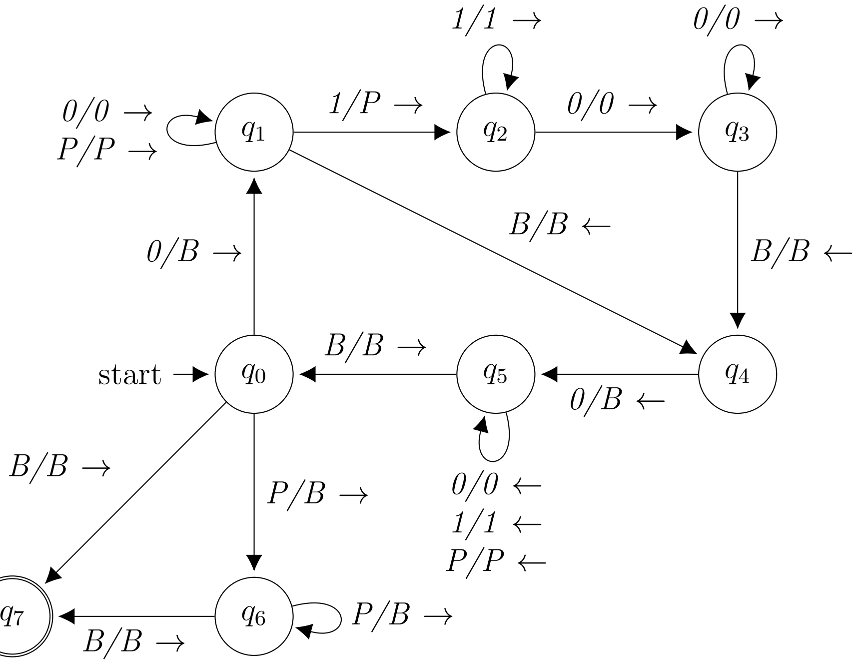
Para a entrada bba, teremos a seguinte sequência de descrições instantâneas: $q_0bba \vdash q_1Bbba \vdash Mq_2bba \vdash q_4MPba \vdash q_4BMPba \vdash bq_5MPba \vdash bMq_2Pba \vdash bMPq_2ba \vdash bMq_4PPa \vdash bq_4MPPa \vdash q_4bMPPa \vdash q_4BbMPPa \vdash bq_5bMPPa \vdash bbq_5MPPa \vdash bbMq_2PPa \vdash bbMPq_2Pa \vdash bbMPPq_2a \vdash bbMPq_3PP \vdash bbMq_3PPP \vdash bbq_3MPPP \vdash bq_3bMPPP \vdash q_3bbMPPP \vdash q_3BbbMPPP \vdash aq_5bbMPPP \vdash abq_5bMPPP \vdash abbq_5MPPP \vdash abbMq_2PPP \vdash abbMPq_2PP \vdash abbMPPq_2P \vdash abbMPPPq_2B \vdash abbMPPq_6P \vdash abbMPq_6P \vdash abbMq_6P \vdash abbq_6M \vdash abq_7b$

Como podemos ver, o traço de computação é algo longo, mesmo para uma cadeia relativamente curta, dado que são necessárias várias passagens para fazer a cópia da cadeia.

Exercício 5 (TPC 2013/14)

Considere a linguagem $L = \{0^m 1^n 0^m, m \geq 0, n \leq m\}$. Desenhe uma Máquina de Turing capaz de reconhecer a linguagem, apresentando também a sua representação formal e tabular. Apresente ainda o traço de computação para a cadeia 00100.

Para reconhecer esta linguagem, podemos usar uma estratégia semelhante à vista acima para a linguagem $a^n b^n c^n, n \geq 1$, percorrendo a cadeia e marcando a cada iteração os zeros e uns processados. Como $n \leq m$, tem de ser possível percorrer a cadeia sem ler qualquer 1 no meio. Por outro lado, temos de ter também em atenção que a cadeia vazia pode ser aceite.



Como podemos ver, começando em q_0, se a cadeia for vazia, transita diretamente para q_7, aceitando. No caso de existirem zeros, o primeiro é eliminado, prosseguindo-se com a leitura da cadeia. Caso a cadeia tenha apenas zeros, a TM transitando diretamente de q_1 para q_4 ao chegar ao final da cadeia, apagando o último zero e regressando ao início da cadeia, voltando a q_0. No caso de existirem uns, é usado o ciclo exterior, marcando o primeiro um como processado (símbolo P), e passando pelos uns e zeros até ao final da cadeia. Tendo as duas opções em q_1, a TM irá processar os uns enquanto estes existirem, pelo ciclo externo, usando depois o ciclo interno para processar os restantes zeros. No final, quando já não existirem zeros no início da cadeia, a transição para q_6 e a leitura de símbolos P irá assegurar que a cadeia está efetivamente no formato especificado, garantindo que não sobraram 1's por processar e que não existem 0's no meio dos 1's.

Usando a notação formal:
$TM = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}, \{0, 1\}, \{0, 1, P, B\}, \delta, q_0, B, \{q_7\})$, sendo δ definido da seguinte forma:
$\delta(q_0, 0) = (q_1, B, R)$; $\delta(q_0, P) = (q_6, B, R)$; $\delta(q_0, B) = (q_7, B, R)$; $\delta(q_1, 0) = (q_1, 0, R)$; $\delta(q_1, P) = (q_1, P, R)$; $\delta(q_1, 1) = (q_2, P, R)$; $\delta(q_1, B) = (q_4, B, L)$; $\delta(q_2, 1) = (q_2, 1, R)$; $\delta(q_2, 0) = (q_3, 0, R)$; $\delta(q_3, 0) = (q_3, 0, R)$; $\delta(q_3, B) = (q_4, B, L)$; $\delta(q_4, 0) = (q_5, B, L)$; $\delta(q_5, 0) = (q_5, 0, L)$; $\delta(q_5, 1) = (q_5, 1, L)$; $\delta(q_5, P) = (q_5, P, L)$; $\delta(q_5, B) = (q_0, B, R)$; $\delta(q_6, P) = (q_6, B, R)$; $\delta(q_6, B) = (q_7, B, R)$

Em forma tabular:

	0	1	P	B
$\rightarrow q_0$	(q_1, B, R)		(q_6, B, R)	(q_7, B, R)
q_1	$(q_1, 0, R)$	(q_2, P, R)	(q_1, P, R)	(q_4, B, L)
q_2	$(q_3, 0, R)$	$(q_2, 1, R)$		
q_3	$(q_3, 0, R)$			(q_4, B, L)
q_4	(q_5, B, L)			
q_5	$(q_5, 0, L)$	$(q_5, 1, L)$	(q_5, P, L)	(q_0, B, R)
q_6			(q_6, B, R)	(q_7, B, R)
$*q_7$				

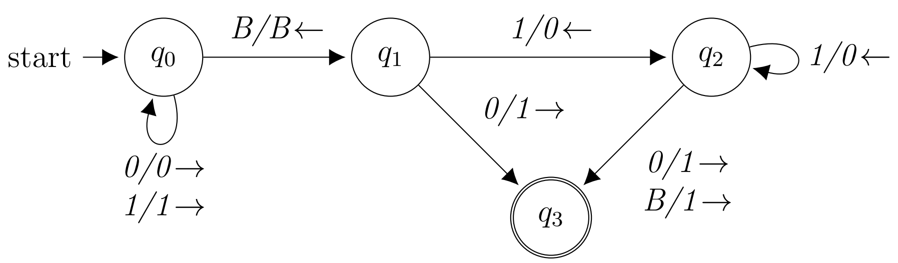
Relativamente ao traço de computação para a cadeia 00100, temos então a seguinte sequência:
$q_0 00100 \vdash q_1 0100 \vdash 0q_1 100 \vdash 0Pq_2 00 \vdash 0P0q_3 0 \vdash 0P00q_3 B \vdash 0P0q_4 0 \vdash 0Pq_5 0 \vdash 0q_5 P0 \vdash q_5 0P0 \vdash q_5 B0P0 \vdash q_0 0P0 \vdash q_1 P0 \vdash Pq_1 0 \vdash P0q_1 B \vdash Pq_4 0 \vdash q_5 P \vdash q_5 BP \vdash q_0 P \vdash q_6 B \vdash q_7 B$

Exercício 6 (Exercício 2015/16)

Pretene-se obter uma Máquina de Turing que adicione 1 ao número inteiro representado em binário na fita. Exemplos: 0010 $\rightarrow$ 0011 ; 1 $\rightarrow$ 10
a) Desenhe uma Máquina de Turing correspondente.
b) Apresente o traço de computação quando a entrada na fita é 101.

Para resolver este problema, podemos pensar numa forma de somar 1 em binário, o que pode ser conseguido percorrendo o número da direita para a esquerda e:
- caso o primeiro bit (bit mais à direita) seja 0, mudar para 1
- caso o primeiro bit seja 1, mudar para 0 e continuar para a esquerda, mudando todos os 1s para 0s até aparecer o primeiro 0 (ou um espaço branco à esquerda do número), o qual se muda para 1.

Desenhando então uma TM que use esta estratégia:



Como podemos ver, em q_0 a cadeia é percorrida para a direita até ao final, sendo em q_1 tomada a decisão sobre qual das opções tomar: caso seja um 0, troca-se para 1, passando diretamente para q_3; caso seja 1, passa-se para q_2, continuando a trocar os 1's até aparecer o primeiro 0 ou uma casa vazia, transitando nesse momento para q_3.

O taço de computação para a cadeia 101 será então:

$q_0 101 \vdash 1q_0 01 \vdash 10q_0 1 \vdash 101q_0 B \vdash 10q_1 1 \vdash 1q_2 00 \vdash 11q_3 0$