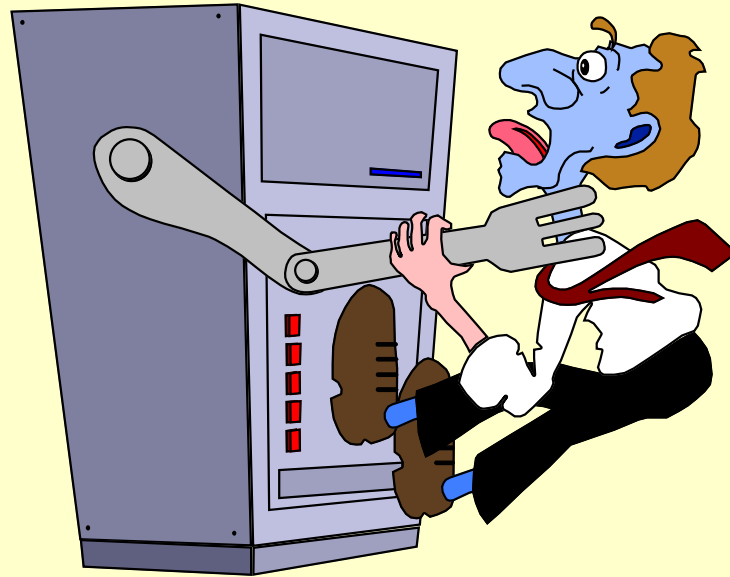


Máquinas de Turing



Máquinas de Turing podem fazer tudo o que um computador real faz.

Porém, mesmo uma Máquina de Turing **não** pode resolver certos problemas. Estes problemas estão além dos limites teóricos da computação

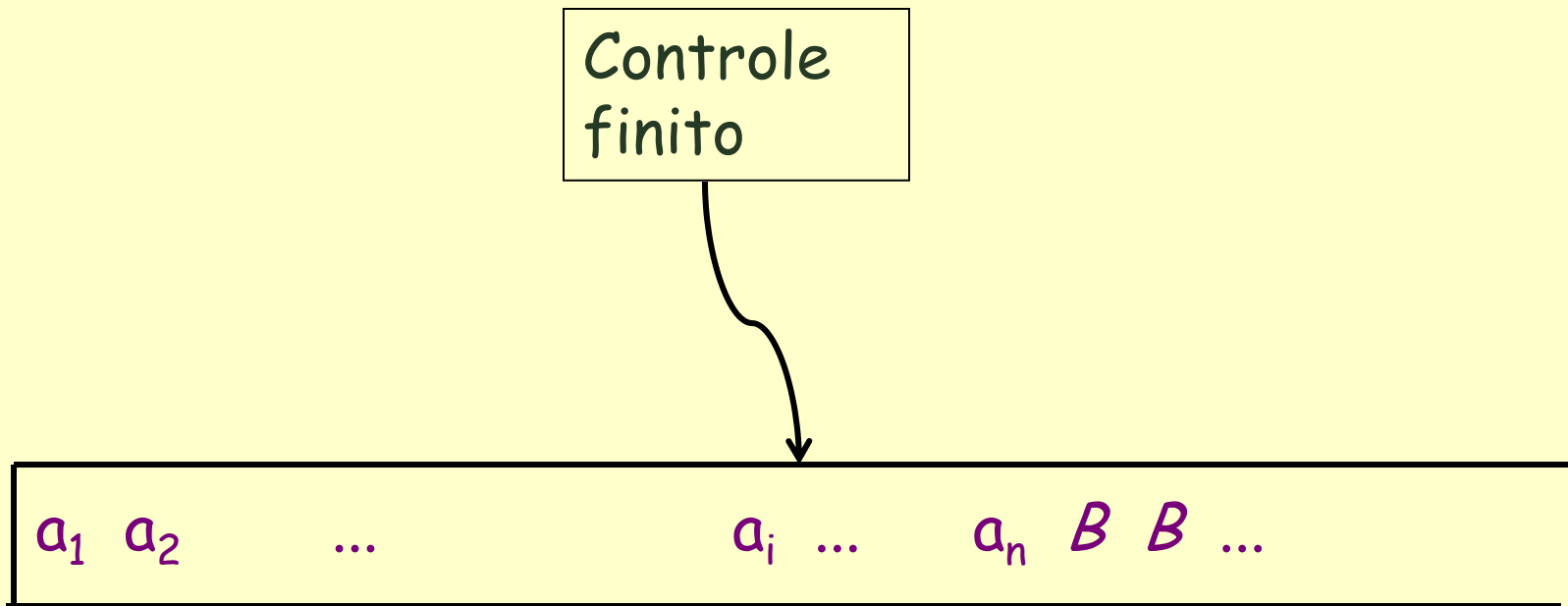
História

- Turing (1936): Máquinas de Turing como modelo de função computável.
- Tese de Church-Turing: qualquer modelo geral de computação permite calcular as mesmas funções (ou, tudo o que se pode computar coincide com as linguagens reconhecidas pelas Máquinas de Turing).

Máquina de Turing

- É uma máquina de estados finitos, M , que reconhece uma linguagem, a $L(M)$
- Ou seja, dada uma cadeia, após uma sequência de mudanças de estados, a M eventualmente para com a resposta *SIM*, se a cadeia pertence à $L(M)$, ou *NÃO*, se não pertence.

Máquina de Turing



Inicialmente, a entrada é colocada na fita infinita. Todas as outras células têm um símbolo especial da fita, B (*branco*).

A cabeça da fita fica posicionada em uma das células. No início, a cabeça está posicionada na célula mais à esquerda que contém a entrada.

Um *movimento* da MT é uma *função* do *estado* do controle finito e do *símbolo atual* da fita. Em um movimento, a MT:

1. Mudará de estado (opcionalmente para o mesmo).
2. Gravará um símbolo de fita na célula atual, substituindo o existente (podendo ser o mesmo).
3. Movimentará (necessariamente) a cabeça da fita uma célula à esquerda ou à direita.

- Uma vez iniciados os movimentos, a MT só para se não houver um movimento previsto, ou seja, se para o estado e o símbolo atuais, não há um movimento previsto.
- Para indicar a **aceitação** de uma cadeia, quando a MT parar, ela deve estar num **estado de aceitação - ou estado final**.
- Se, ao parar, a MT não estiver num estado final, então a MT **rejeita** a cadeia de entrada.

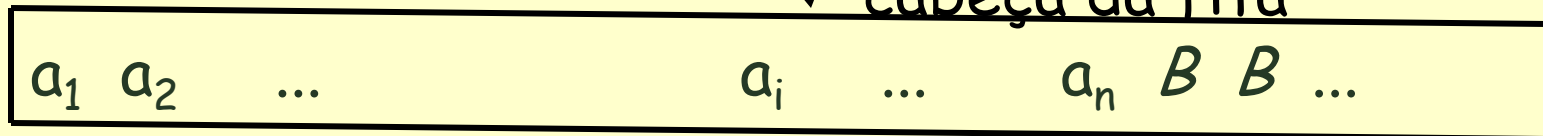
MT: notação formal

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

Controle
finito

Q = conj. finito de estados;
 F = conj. estados finais (de aceitação)

Γ = alfabeto finito da fita



Σ = alfabeto finito de entrada

Máquina de Turing

Os movimentos da MT são definidos por uma Função de transição δ :

$$\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\}$$

Ou seja, $\delta(q,a) = (p,b,D)$ onde:

- p é o próximo estado em Q ;
- b é o símbolo que substituirá a na fita;
- D é uma direção (esquerda ou direita) em que a cabeça da fita irá se mover.

A linguagem de uma MT

- **Intuitivamente:** a cadeia de entrada é colocada na fita, e a cabeça da fita começa no símbolo mais à esquerda da cadeia. Se a MT parar eventualmente num estado de aceitação (de F), a entrada é dita aceita ou reconhecida; caso contrário, não.
- **Assim**, a **Linguagem $L(M)$ Aceita ou Reconhecida** por uma MT, $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$, é o conjunto de cadeias $w = w_1w_2\dots w_n$ em Σ^* tais que $\delta(q_0, w_1)$ leva a uma sequência de transições que culminem num estado p de F (**aceitação por estado final**). Não importa o conteúdo final da fita neste caso!

A linguagem de uma MT

- As linguagens aceitas por MT são também chamadas de *linguagens recursivamente enumeráveis (LRE)*
- Pela Tese de Church, as funções computáveis coincidem com as LRE

MT e sua parada

- Há uma outra noção de “aceitação” para MT: a **aceitação por parada**. Em geral, usada quando o conteúdo final da fita representa alguma resposta ao problema que a MT representa.
- Dizemos que uma MT **para** se ela entra em um estado **q**, olhando um símbolo de fita **a**, e não existe qualquer movimento previsto nessa situação, i.e., **$\delta(q,a)$ é indefinida**. Não se define o conjunto de estados finais F nesse caso.

Usos de uma MT

- como reconhecedor de linguagens
- para calcular funções (resolver problemas)

Usos de uma MT

- como reconhecedor de linguagens

(A linguagem corresponde ao conjunto de instâncias que geram resposta SIM ao problema de decisão associado: *esta cadeia pertence à linguagem $L(M)$? ou esta é uma solução do problema?*)

- para calcular funções
(equivale ao problema propriamente dito)

Exemplo

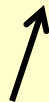
- Vamos projetar uma MT para reconhecer

$$L = \{0^n 1^n \mid n \geq 1\}$$

Ou seja, para toda entrada de cadeias de 0s seguidos de 1s, em igual número, a MT deve parar num estado final. Para as demais cadeias binárias, deve parar num estado não final.

Ex.: 01, 0011, 000111 fazem parar num estado final;
1, 10, 001, 1010 fazem parar num estado não final.

Fita inicial: 000...111BBBBBB

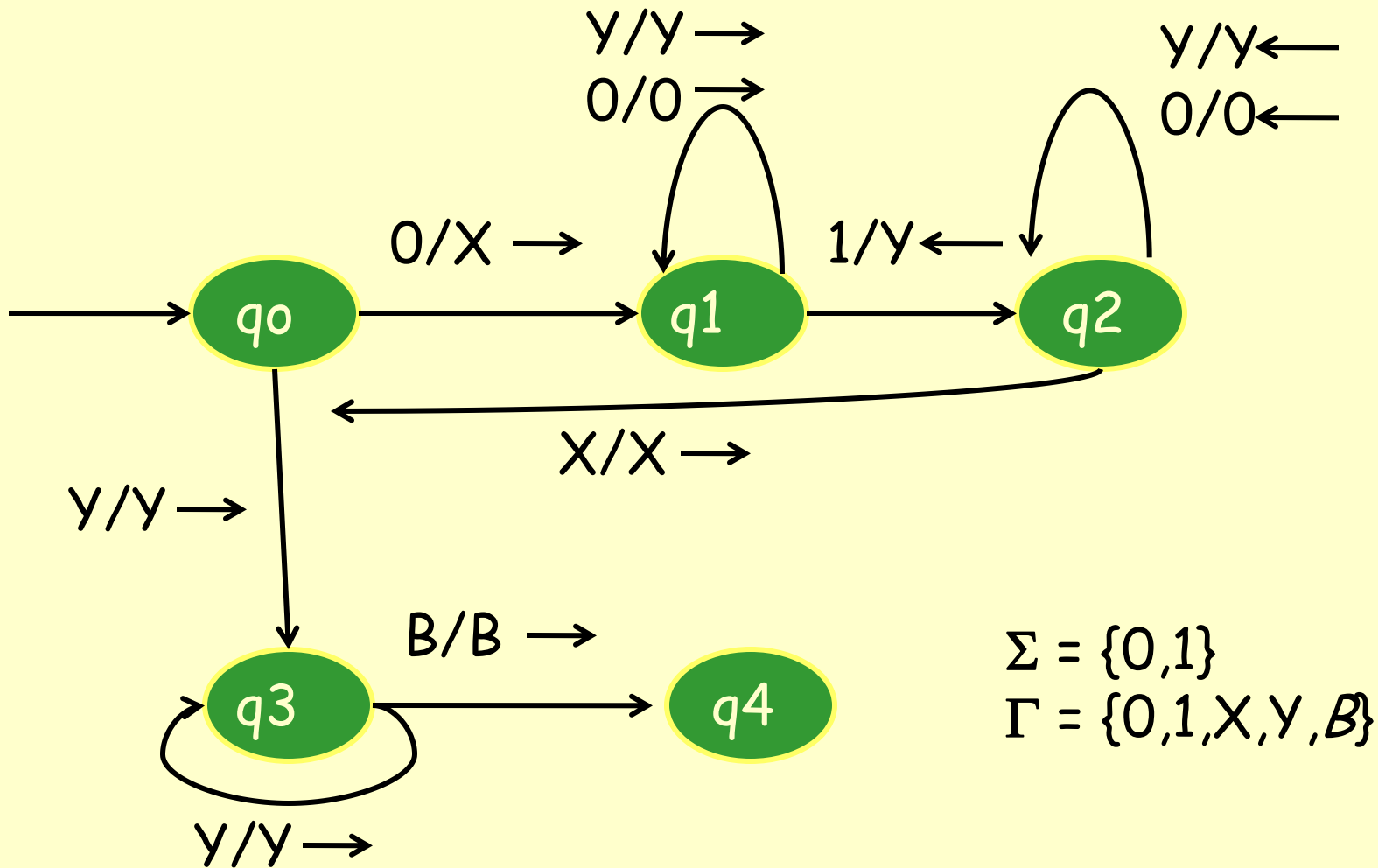


- **Estratégia:** em cada passo, a MT trocará um 0 por um X, e depois um 1 por um Y, até todos os 0s e 1s terem sido trocados aos pares.

Exemplo

- **Estratégia:** em cada passo, a MT trocará um 0 por um X, e depois um 1 por um Y, até todos os 0s e 1s terem sido trocados aos pares.
- **Em cada passo**, da esq. para dir., ela troca um 0 por X e vai para a direita, ignorando 0s e Ys até encontrar 1. Troca esse 1 por Y e se move para a esquerda, ignorando Ys e 0s, até encontrar um X. Procura um 0 a direita e troca por X, repetindo o processo.
- **Se a entrada não for da forma $0^n 1^n$** eventualmente a MT não vai ter um movimento previsto (previmos apenas os movimentos para cadeias válidas) e **vai parar sem aceitar**, ou seja, num estado que não é de F.
- Se, por outro lado, na busca por mais um 0, ela **só encontrar Xs e Ys**, então ela descobre que **deve aceitar** a entrada, e vai para um estado final.

Diagrama de Transição



$$M = (\{q_0, q_1, q_2, q_3, q_4\}, \{0, 1\}, \{0, 1, X, Y, B\}, \delta, q_0, \{q_4\})$$

Estado	0	1	X	Y	B
→ q ₀	(q ₁ , X, R)	--	--	(q ₃ , Y, R)	--
q ₁	(q ₁ , 0, R)	(q ₂ , Y, L)	--	(q ₁ , Y, R)	--
q ₂	(q ₂ , 0, L)	--	(q ₀ , X, R)	(q ₂ , Y, L)	--
q ₃	--	--	--	(q ₃ , Y, R)	(q ₄ , B, R)
q ₄ *	--	--	--	--	--

Verifique se a cadeia 000111 é aceita

- Qual é a complexidade dessa MT (desse algoritmo)?

- Qual é a complexidade dessa MT (desse algoritmo)?
- Se n é o tamanho da cadeia de entrada;
- (a) Em cada passo, busca-se um 0 e um 1 na cadeia, portanto, no máximo todos os símbolos da cadeia são lidos;
- (b) O número de passos é função do número de pares de 0 e 1, ou seja, do tamanho da cadeia.
- Logo, (a) * (b) = $O(n^2)$

Exercício

- Construa uma MT para reconhecer cadeias de $L = \{w\#w \mid w \in \{0,1\}^*\}$

Estágios para a resolução:

- Verifique (em zigue-zague) se antes e depois do # existem os mesmos símbolos, cc rejeite.
- Ao checar um símbolo, marque-o (use um X por exemplo) para ter controle sobre os que estão sendo analisados num dado momento.
- Quando todos os da esquerda forem checados (com X) verifique se existe algum símbolo à direita ainda não checado. Se houver, rejeite; cc aceite.
- Complexidade: Por raciocínio análogo ao anterior, $O(n^2)$

MT como um processador de funções inteiras

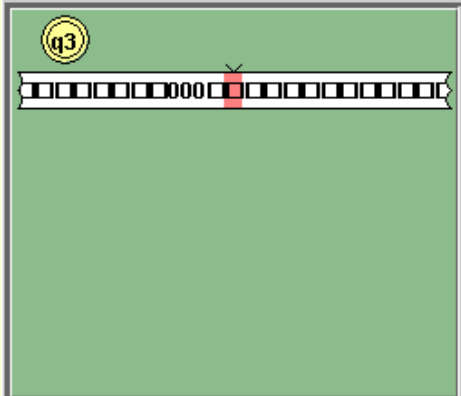
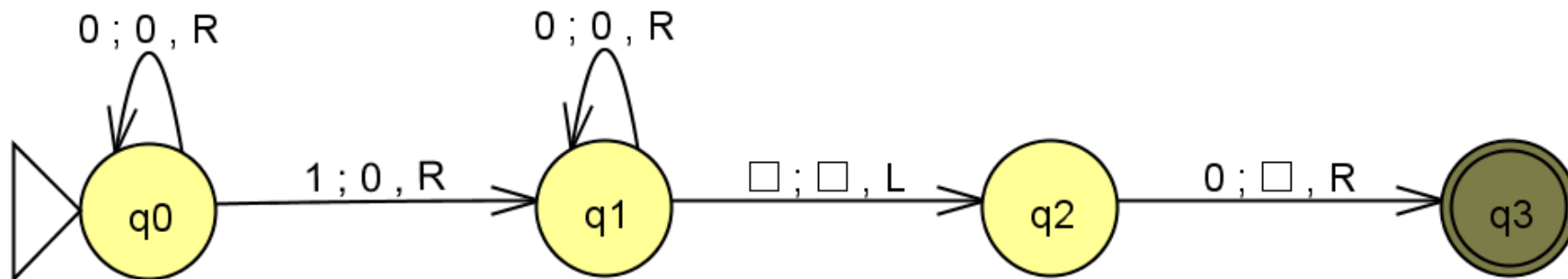
- Tradicionalmente, os inteiros são representados em vocabulário unário.
- O inteiro $i \geq 0$ é representado pela cadeia 0^i .
- Se a função tem k argumentos (i_1, i_2, \dots, i_k) então esses inteiros são colocados na fita separados por 1's como:

$0^{i_1} 1 0^{i_2} 1 \dots 1 0^{i_k}$

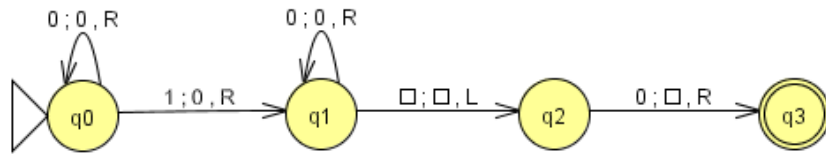
- O inverso também é possível.
- Se a máquina pára (não importa em que estado) com a fita consistindo de 0^m para algum m , então dizemos que $f(i_1, i_2, \dots, i_k) = m$, onde f é uma função de k argumentos computados por essa MT.

Exemplo: MT que soma dois números naturais, $a + b$

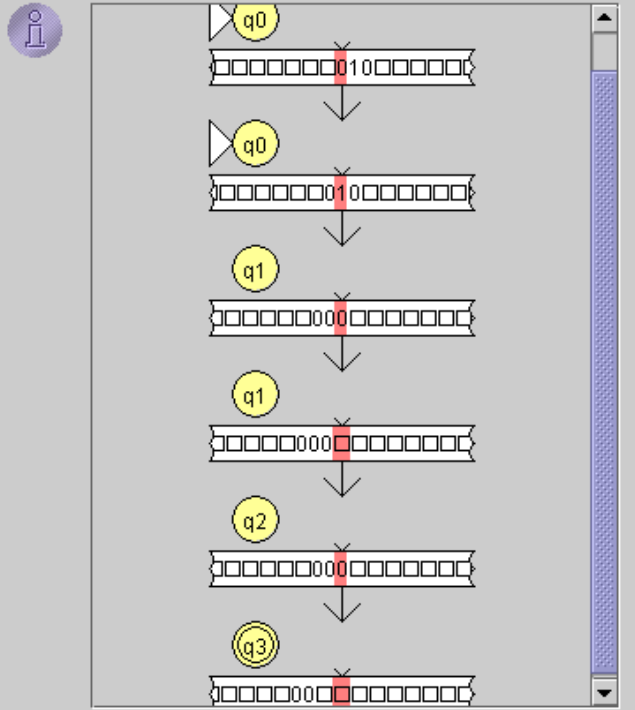
- Conteúdo inicial da Fita: $0^a 1 0^b B...$
- Quando a MT parar, o conteúdo da fita dever ser:
 $0^{a+b} B....$
- **Processo:**
- Ler o 0 mais à esquerda, mantendo-o como 0, e mover à direita até encontrar o 1.
- Substitua o 1 por 0 (nesse momento a cadeia da fita é 0^{a+b+1}). Continue movendo à direita sem mudar a fita, até que um B seja encontrado.
- Mantenha o B e mova a esquerda para encontrar o último 0 mais a direita.
- Substitua esse 0 por B . O resultado é $B 0^{a+b}$
- Qual é a complexidade desse algoritmo?



Editor



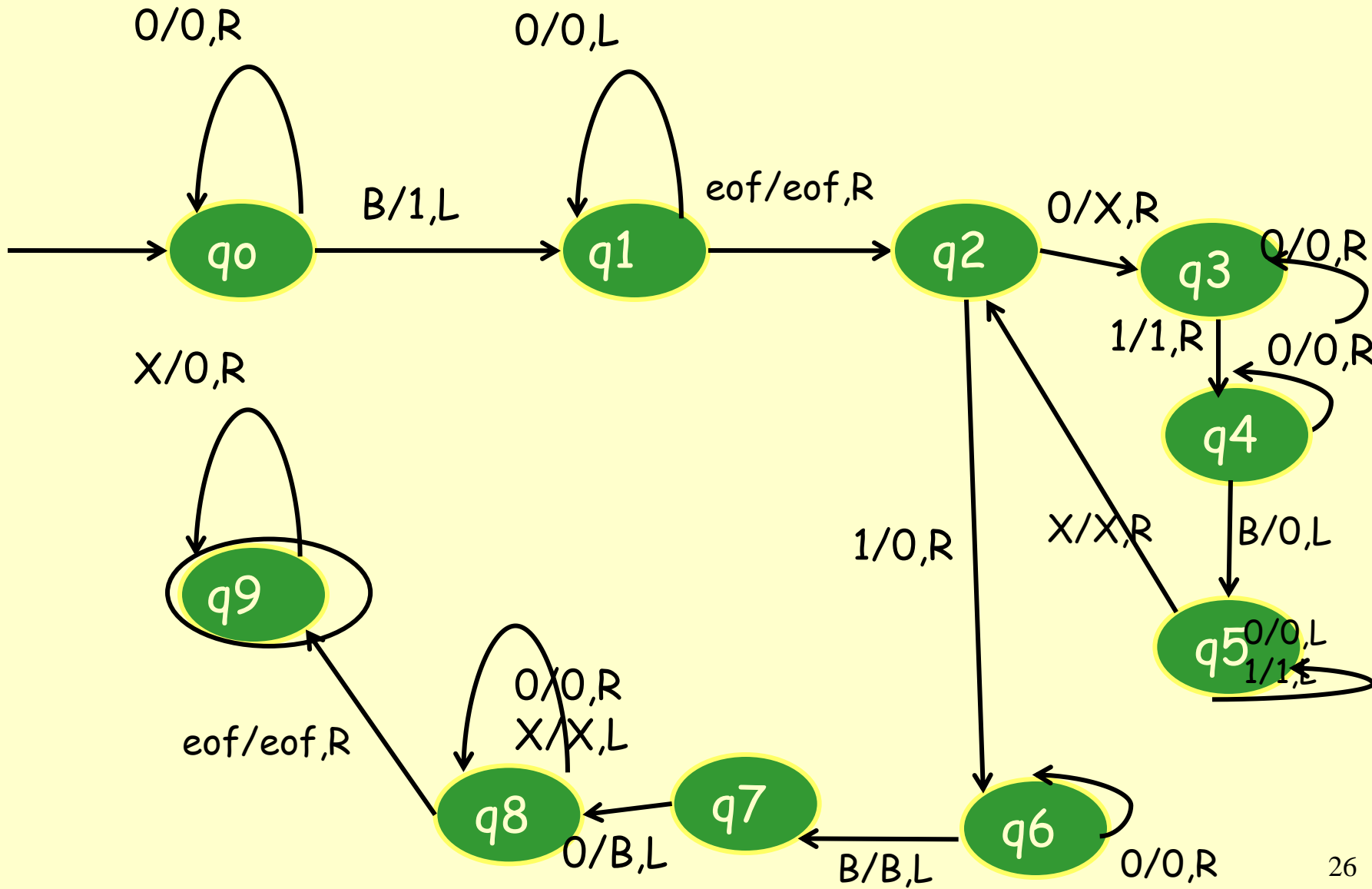
Accepting configuration found!



Keep looking I'm done

Exemplo: MT que multiplica um número natural por 2- $a * 2$

- Conteúdo inicial da Fita: $0^a B...$
- Quando a MT parar, o conteúdo da fita dever ser:
 $0^{a+a} B....$
- **Sugestão:**
- Grave 1 depois do último 0.
- Grave o mesmo número de 0s da entrada, à direita do 1. Haverá necessidade de substituir os 0s originais por X, p.ex.
- Substitua o 1 por 0 (nesse momento a cadeia da fita é $X^a 0^{a+1}$). Continue movendo à direita sem mudar a fita, até que um B seja encontrado.
- Mantenha o B e mova a esquerda para encontrar o último 0 mais a direita. Substitua esse 0 por B, para obter $X^a 0^a$.
- Substitua todos os X por 0. O resultado é 0^{a+a}



- F é vazio se a MT computa uma função.
- F é relevante quando a MT é usada para reconhecer uma linguagem.

Ex. Uma MT para reconhecer a Linguagem

$$L = \{ a^n b^n c^n \mid n \geq 0 \}$$

Exemplos:

Pertence à L:

aaabbbccc

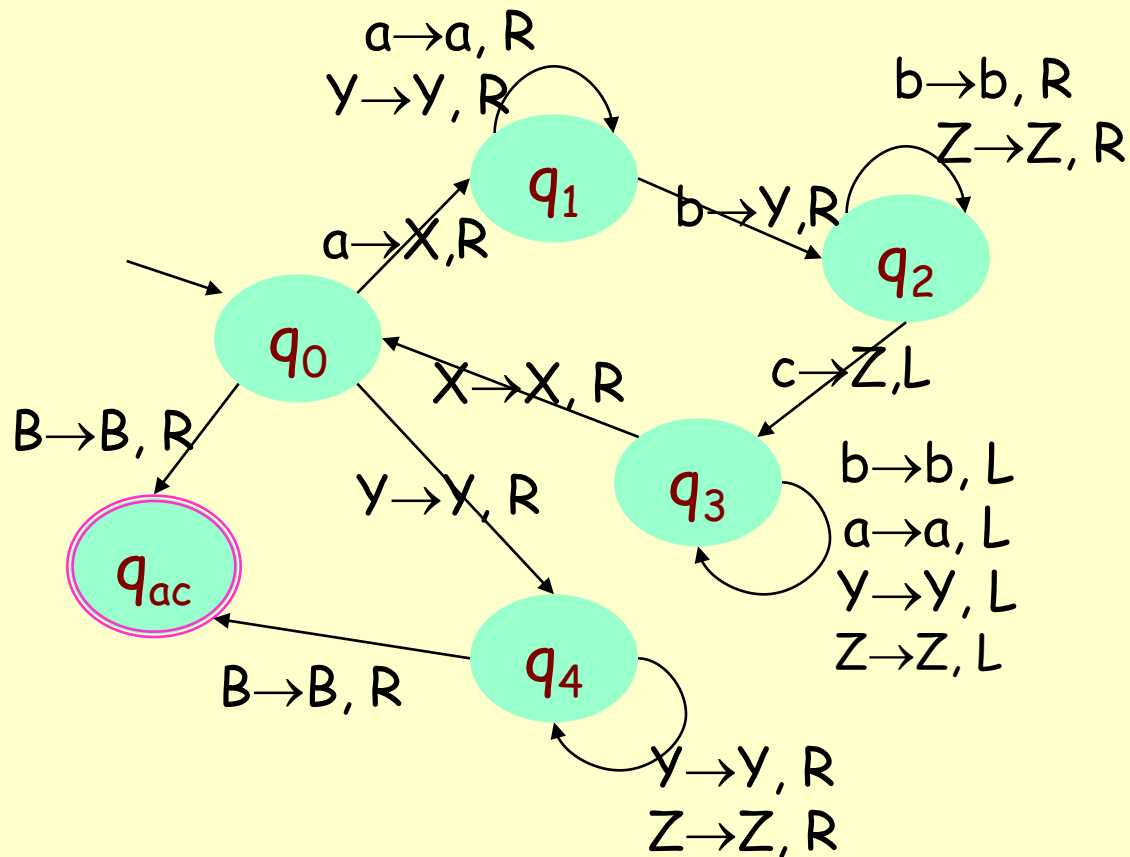
Não Pertence à L:

aaabbbccc

A Máquina de Turing

1. $Q = \{q_0, q_1, q_2, q_3, q_4, q_{ac}\}$
2. $\Sigma = \{a, b, c\}$
3. $\Gamma = \{a, b, c, B, X, Y, Z\}$
4. δ a seguir.
5. q_0 - o estado inicial
6. $F = \{q_{ac}\}$

Ideia: em cada passo, reconhecer um a, um b e um c, substituindo-os por X, Y e Z, respectivamente.



Exercícios

- 1) Construir uma MT que decida se uma sequência de parênteses é bem formada.
 - Dica: considere que a cadeia de parênteses é limitada por 2 A's (um a esq. e outro à direita).
 - Ideia: Procure por um $)$ e substitua por X ; em seguida, volte à esquerda procurando o $($ mais próximo para substituir por X também.
- 2) Construir uma MT tal que, dada uma cadeia w pertencente a $\{0,1\}^*$, duplique w . Quando a máquina parar, a fita deve conter $w\#w$ sendo que $\#$ indica fim de w .

Exercício

Faça uma MT que reconheça $L = \{x \mid x \in \{a,b,c\}^* \text{ e } x \text{ é uma permutação de } a^n b^n c^n \text{ para algum } n \geq 0\}$

Exs.: aabbcc ✓

bca ✓

cccaaabbb ✓

babacc ✗

aabccc ✗

aacc ✗

Sugestão

a) trocar um a,b, ou c do começo por 1 para marcar o final à esquerda;

b) substituir um a, um b e um c por 0's.

c) M aceita se, ao percorrer a cadeia de entrada, a fita consiste somente de 0's.