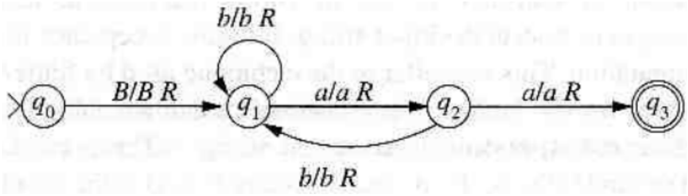
Algoritmos e Fundamentos da Teoria de Computação

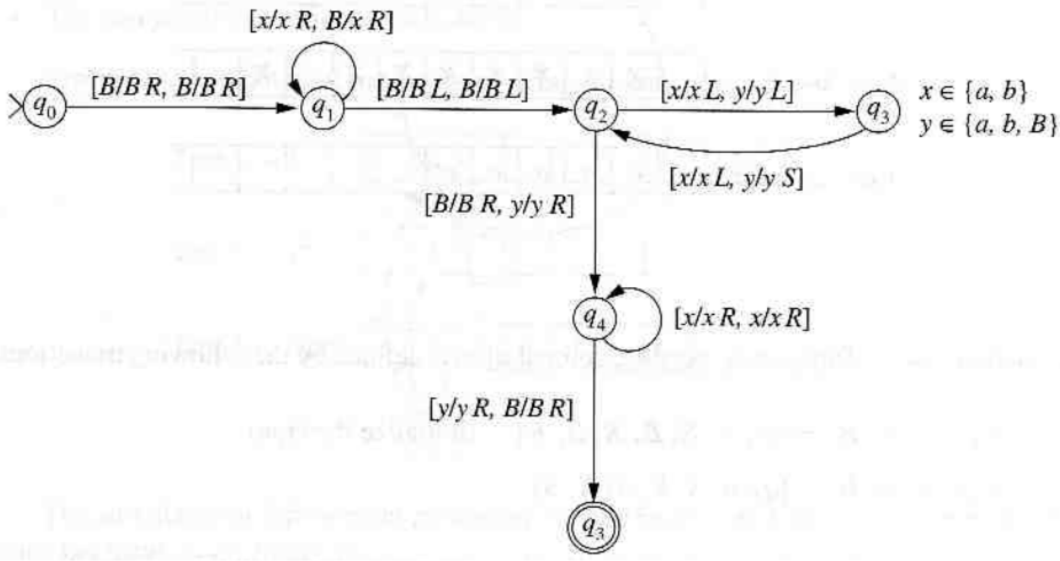
Lista de Exercícios 05

1 Determine a complexidade de tempo das seguintes DTMs. Calcule a fórmula completa da função tc_{M} para cada item.

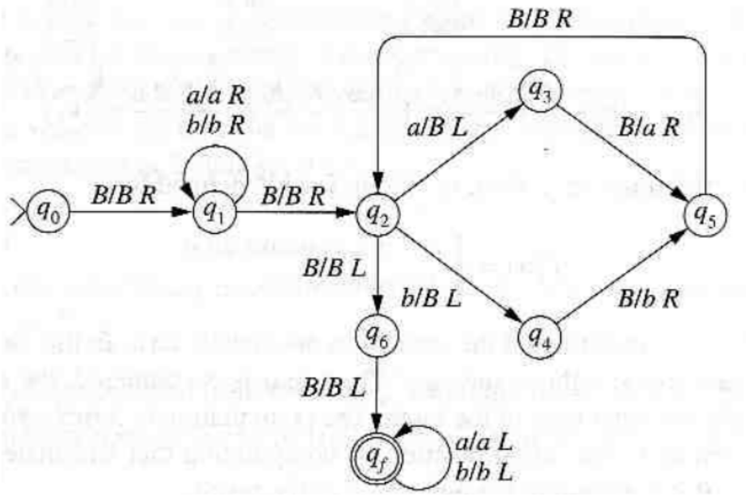
a. DTM padrão que aceita a linguagem $\mathsf{L} = (\boldsymbol{a} \cup \boldsymbol{b})^*\boldsymbol{aa}(\boldsymbol{a} \cup \boldsymbol{b})^*$.



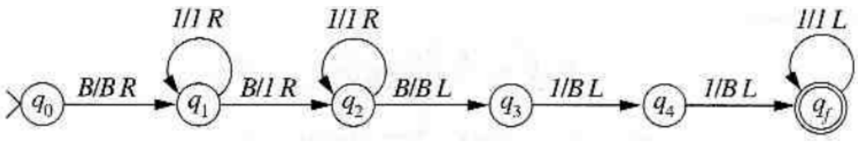
b. DTM duas-fitas que aceita a linguagem $\mathsf{L} = \{uu \mid u \in \{a, b\}^*\}$.



c. DTM que computa a função de concatenação de *strings* sobre $\{a, b\}$.



d. DTM que computa a função de soma de dois números naturais em notação unária.



a. O pior caso acontece quando a máquina aceita *strings* que terminam exatamente com aa (sem sufixo). Nesse caso, se a *string* de entrada tem tamanho n, a máquina percorre toda a *string* para aceitá-la, e, portanto, $tc_{\mathsf{M}} = n + 1$.

b. O pior caso acontece quando a máquina aceita a entrada. Considere que a entrada tem tamanho n, isto é, *length* $(uu) = n$. Logo, *length* $(u) = n/2$. A quantidade de transições realizada pela máquina é resumida na tabela abaixo.

Passo	**No. transições**
$q_0 \to q_1$	1
Loop em q_1 (copia uu na fita 2)	n
$q_1 \to q_2$	1
Loop entre q_2 e q_3 (rebobina fita 1)	n
$q_2 \to q_4$	1
Loop em q_4 (compara primeira metade com a segunda)	$n/2$
$q_4 \to q_5$	1

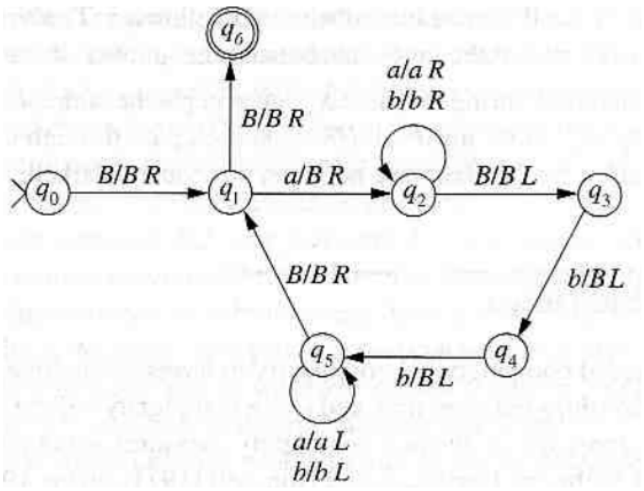
A complexidade de tempo é obtida somando-se as colunas da tabela acima. Logo, $tc_{\mathsf{M}} = \frac{5}{2}n + 4$.

c. O tamanho da entrada em uma máquina de Turing que computa uma função com mais de uma variável é considerado como o tamanho dos argumentos mais os brancos que os separam. A máquina dessa questão realiza a computação da função de concatenação de *strings*. A entrada tem a forma $BuBvB$, com tamanho *length* (u) + *length* $(v) + 1$.

Uma computação percorre u e depois desloca cada elemento de v uma posição para a esquerda. Para uma entrada de tamanho n, o pior caso ocorre quando $u = \lambda$ e *length* $(v) = n - 1$. Uma computação com entrada BBv começa lendo os dois brancos que precedem v. A translação de cada símbolo de v requer três transições. A computação termina com a cabeça retornando para a posição 0 da fita, o que requer *length* $(v) + 2 = n + 1$ transições. Assim, a complexidade de tempo é $tc_{\mathsf{M}} = 2 + 3(n-1) + n + 1 = 4n$.

d. Sejam $x, y \in \mathbf{N}$ em notação unária, então a complexidade da entrada é *length* $(1^{x+1}B1^{y+1}) = x + y + 3$. Para qualquer entrada válida, a máquina sempre realiza a mesma quantidade de transições, isto é, o pior e melhor caso são iguais. Basta ver que a computação sempre avança até o final da entrada e depois retorna à posição inicial. Então, se $x + y + 3 = n$, temos que $tc_{\mathsf{M}} = 2n + 2$.

2 Seja M a DTM ilustrada abaixo. Pede-se:



a. Faça o *trace* da computação de M para as entradas λ, a e abb.

b. Descreva a *string* de tamanho n para a qual a computação de M requer o número máximo de transições.

c. Apresente a função tc_{M}.

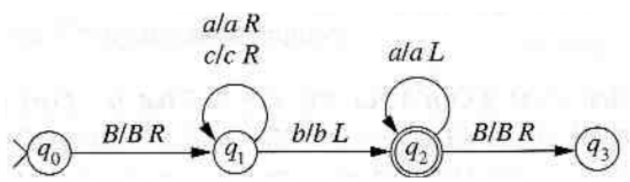
a. Trivial. A máquina aceita λ e abb, e rejeita a.

b. A linguagem aceita pela máquina é $\mathsf{L} = \{a^i b^{2i} \mid i \geq 0\}$ e o pior caso da computação ocorre quando a máquina aceita a entrada, cujo tamanho é *length* $(a^i b^{2i}) = 3i = n$.

c. A cada passo do *loop*, a máquina apaga um a e dois b's. Inicialmente, a entrada tem tamanho $3i$ e portanto M realiza $3i+1$ transições caminhando para a direita. Ao retornar caminhando para a esquerda, M volta até a posição que continha um a, fazendo mais $3i$ transições. No segundo passo do *loop*, temos que o tamanho da *string* diminuiu de 3 símbolos, sendo agora $3i - 3 = 3(i - 1)$. O *loop* termina quando a *string* for totalmente apagada. Assim, a função de complexidade de tempo para uma entrada de tamanho $3i = n$ é dada por

$$tc_{\mathsf{M}}(n) = \sum_{k=0}^{i}(6k+1) + 2 = (i+1)(3i+1) + 2 = 3i^2 + 4i + 1 + 2 = \frac{n^2}{3} + \frac{4n}{3} + 3 \quad .$$

3 Seja M a DTM ilustrada abaixo. Pede-se:



a. Faça o *trace* da computação de M para as entradas abc, aab e cab.

b. Descreva a *string* de tamanho n para a qual a computação de M requer o número máximo de transições.

c. Apresente a função tc_{M}.

a. Trivial. A máquina rejeita abc e aab, e aceita cab.

b. A linguagem aceita por M é regular e pode ser descrita como $\mathsf{L} = (\boldsymbol{a} \cup \boldsymbol{c})^* \boldsymbol{c}\boldsymbol{a}^* \boldsymbol{b} (\boldsymbol{a} \cup \boldsymbol{b} \cup \boldsymbol{c})^*$. Ao contrário dos exercícios anteriores, o pior caso acontece quando M *rejeita* entradas da forma a^*b. Nesse caso, se *length* $(a^*b) = n$, a máquina varre a *string* completa duas vezes.

c. Dado a explicação do item anterior, temos que $tc_{\mathsf{M}}(n) = 2n + 1$.

4 Seja a linguagem $\mathsf{L} = \{a^i b^i \mid 0 \leq i \leq 50\} \cup \{u \mid \textit{length}(u) > 100\}$ definida sobre o alfabeto $\Sigma = \{a, b\}$.

a. Projete uma DTM padrão M que aceita L. (Apresente M como um algoritmo.)

b. Apresente a função tc_{M}.

c. Qual é a melhor notação assintótica que descreve o crescimento de tc_{M}?

a. A computação de uma TM padrão que aceita a linguagem pedida começa utilizando 100 estados para verificar o tamanho da entrada. Se a máquina não encontrar um branco após 100 transições, a computação termina e aceita a entrada.

   Se a TM ler um branco antes de percorrer 100 símbolos da entrada, a computação

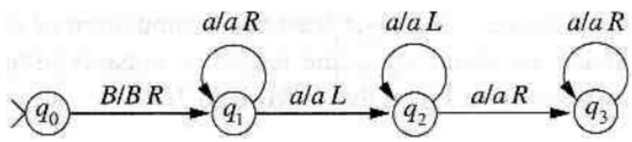
   1. move a cabeça da fita para a posição 0, e
   2. executa uma máquina que aceita $\{a^i b^i \mid i \geq 0\}$.

b. O passo 2 do algoritmo da TM pode ser realizado em $(n+2)(n+1)/2$ transições para uma *string* de entrada de tamanho n. Assim, o número máximo de transições que a TM realiza é dado por

$$tc_{\mathsf{M}}(n) = \begin{cases} 2n + 2 + (n+2)(n+1)/2 & \text{se } n \leq 100 \\ 101 & \text{caso contrário.} \end{cases}$$

c. A complexidade assintótica da computação é constante, pois o número de transições de M é constante quando n tende ao infinito. Assim, $tc_{\mathsf{M}}(n) \in O(1)$.

5 Seja M a NTM ilustrada abaixo. Pede-se:



a. Apresente os *traces* para as computações de M para a entrada aa.

b. Descreva a computação de M para entrada a^n que requer o número máximo de transições.

c. Apresente a função tc_{M}.

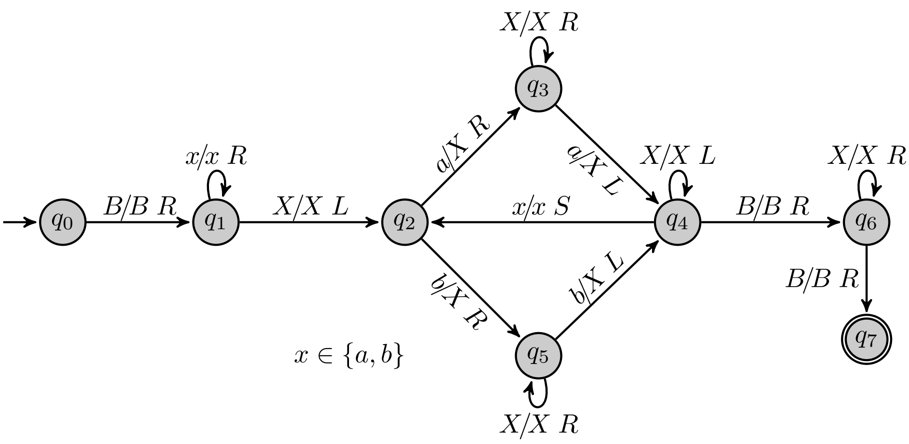
a. As computações de M para a entrada aa são listadas abaixo.

q_0BaaB	q_0BaaB	q_0BaaB	q_0BaaB
$\vdash Bq_1aaB$	$\vdash Bq_1aaB$	$\vdash Bq_1aaB$	$\vdash Bq_1aaB$
$\vdash Baq_1aB$	$\vdash Baq_1aB$	$\vdash Baq_1aB$	$\vdash q_2BaaB$
$\vdash Baaq_1B$	$\vdash Bq_2aaB$	$\vdash Bq_2aaB$	
	$\vdash q_2BaaB$	$\vdash Baq_3aB$	
		$\vdash Baaq_3B$	

b. O desempenho de pior caso para a entrada a^n ocorre quando a máquina lê os primeiros $n-1$ a's no estado q_1 e depois passa para q_2, andando para a esquerda até ler o primeiro a, e finalmente, anda para a direita em q_3 até a computação terminar.

c. Para $n > 0$, a função de complexidade de tempo de M é $tc_{\mathsf{M}}(n) = 3n - 1$.

6 A DTM M apresentada abaixo aceita a linguagem $\mathsf{L} = \{uXu^R \mid u \in \{a,b\}^+\}$, aonde u^R é o **reverso** da *string* u. Determine a complexidade **assintótica** de tempo da máquina M, justificando adequadamente a sua resposta. *(Obs.: Não é necessário determinar a fórmula completa da função tc_{M}, somente o seu comportamento assintótico.)*



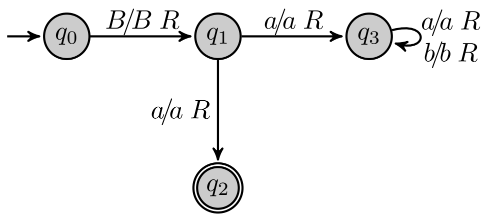
O pior caso da computação da máquina ocorre quando M aceita a entrada, e, nesse caso, sabemos que a *string* é da forma $w = uXu^R$. Seja *length* (u) = *length* $(u^R) = m$. Então, *length* $(w) = n = 2m+1$. É fácil perceber que a maior parte da computação de M ocorre no *loop* que faz um "zig-zag" sobre a entrada. Baseado em todos os exemplos e exercícios vistos na disciplina, podemos de imediato afirmar que a complexidade do *loop* é quadrática em relação ao tamanho da entrada. (As demais partes da máquina contribuem somente um fator linear na complexidade.)

Fazendo uma análise mais detalhada do *loop*, é possível determinar que o número de transições feitas em cada passo cresce segundo a sequência $6, 10, 14, \ldots$, com o passo k realizando $2(2k+1)$ transições. Dado que o *loop* leva m passos, temos que a sua complexidade é dada por

$$\sum_{k=1}^{m} 2(2k+1) = 2m^2 + 4m \in O(m^2).$$

Como $m = \frac{n-1}{2}$, temos que $O(m^2) = O(\frac{(n-1)^2}{4}) \in O(n^2)$, e assim vemos que a complexidade é quadrática em relação ao tamanho da entrada, como esperado.

7 Considere a NTM M abaixo cujo alfabeto de entrada é $\Sigma = \{a,b\}$. Qual é a linguagem aceita por M? Determine a fórmula **completa** da função tc_{M}.



A linguagem aceita por M é $\mathsf{L} = \boldsymbol{a(a \cup b)^*}$. A complexidade de tempo de uma NTM é definida sobre o ramo mais longo da árvore de computações. Como há uma sequência de escolhas de M que sempre percorre a entrada toda, o pior caso de M é linear em relação ao tamanho da entrada, e temos que $tc_{\mathsf{M}}(n) = n + 1$.