Algoritmos e Fundamentos da Teoria de Computação

Lista de Exercícios 01

1 Seja M a máquina de Turing definida pela função δ abaixo.

δ	B	a	b	c
q_0	q_1, B, R			
q_1	q_2, B, L	q_1, a, R	q_1, c, R	q_1, c, R
q_2		q_2, c, L		q_2, b, L

a. Construa o *trace* da computação de M para a *string* de entrada $aabca$.
b. Construa o *trace* da computação de M para a *string* de entrada $bcbc$.
c. Apresente o diagrama de estados de M.
d. Descreva o resultado de uma computação de M.

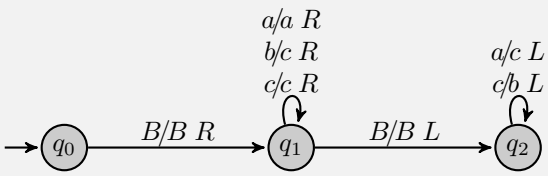
a.

$$
\begin{aligned}
& q_0Baabcab \\
\vdash\ & Bq_1aabcaB \\
\vdash\ & Baq_1abcaB \\
\vdash\ & Baaq_1bcaB \\
\vdash\ & Baacq_1caB \\
\vdash\ & Baaccq_1aB \\
\vdash\ & Baaccaq_1B \\
\vdash\ & Baaccq_2aB \\
\vdash\ & Baacq_2ccB \\
\vdash\ & Baaq_2cbcB \\
\vdash\ & Baq_2abbcB \\
\vdash\ & Bq_2acbbcB \\
\vdash\ & q_2BccbbcB
\end{aligned}
$$

b.

$$
\begin{aligned}
& q_0BbcbcB \\
\vdash\ & Bq_1bcbcB \\
\vdash\ & Bcq_1cbcB \\
\vdash\ & Bccq_1bcB \\
\vdash\ & Bcccq_1cB \\
\vdash\ & Bccccq_1B \\
\vdash\ & Bcccq_2cB \\
\vdash\ & Bccq_2cbB \\
\vdash\ & Bcq_2cbbB \\
\vdash\ & Bq_2cbbbB \\
\vdash\ & q_2BbbbbB
\end{aligned}
$$

c. O diagrama de estados de M é



d. O resultado de uma computação de M é substituir os a's da *string* de entrada por c's e os c's por b's.

2 Construa uma máquina de Turing que realiza as computações pedidas nos itens abaixo. (Faça uma máquina para cada item.) Todas as máquinas têm alfabeto de entrada $\Sigma = \{a, b\}$. Note que a cabeça da máquina deve sempre estar na posição 0 da fita quanto a computação termina no estado q_f.

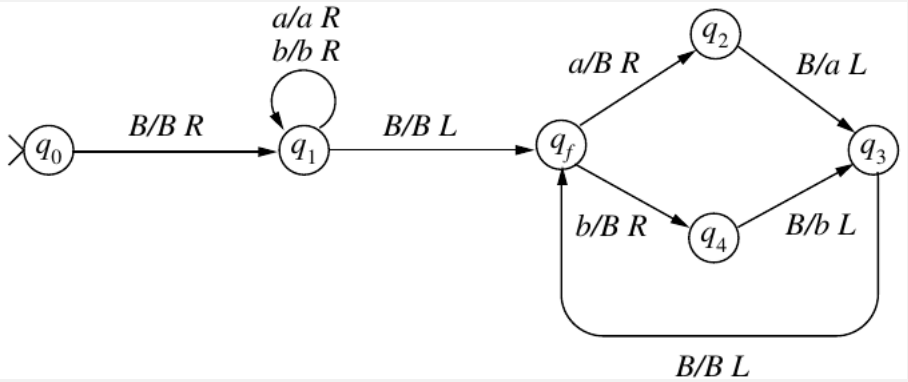
a. Mover a entrada uma posição para a direita: $q_0BuB \models^* q_fBBuB$, aonde $u \in \Sigma^*$.

b. Concatenar uma cópia invertida à *string* de entrada: $q_0BuB \models^* q_fBuu^RB$, aonde u^R é o reverso da *string* u.

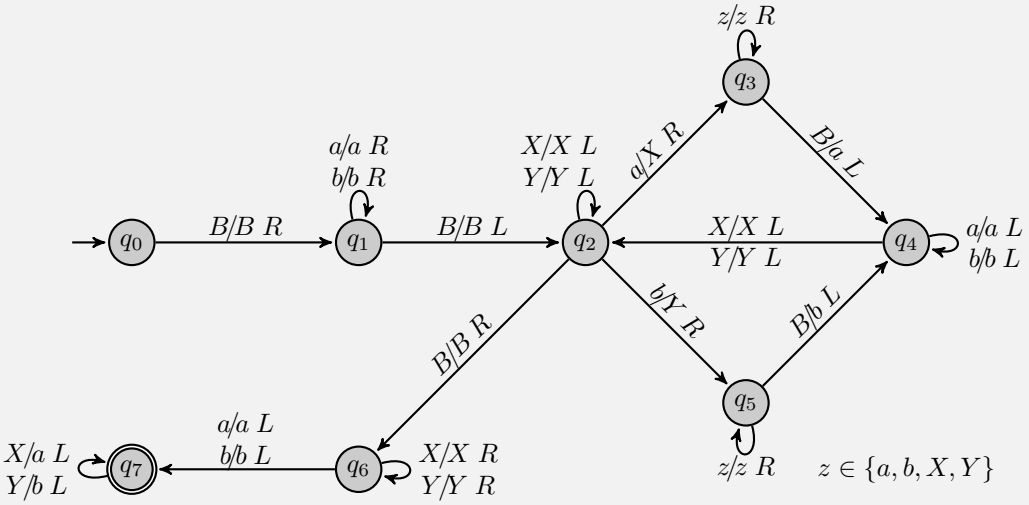
c. Inserir um branco entre cada um dos símbolos da entrada, por exemplo: $q_0BabaB \models^* q_fBaBbBaB$.

d. Apagar os b's da entrada, por exemplo: $q_0Bbabaabab \models^* q_fBaaaaB$.

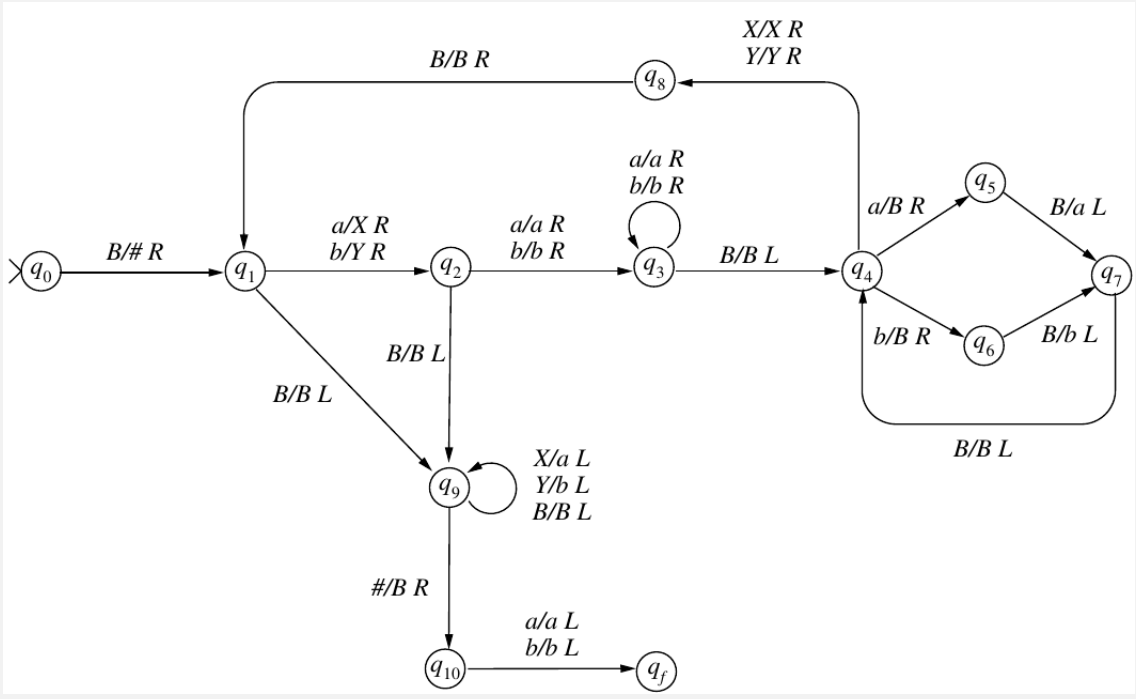
a. A máquina abaixo trabalha da direita para a esquerda, movendo um símbolo de cada vez.



b. A máquina abaixo utiliza os símbolos X e Y para marcar, respectivamente, os símbolos a e b que já foram copiados para a *string* inversa.



c. A máquina abaixo utiliza a máquina do item a) iterativamente para inserir brancos entre os símbolos.



O algoritmo da máquina é o seguinte:

1. Marcar a posição 0 da fita com #.
2. Se a entrada não é vazia, trocar o primeiro símbolo por X ou Y para registrar se era um a ou b, respectivamente.
3. Caminhar até o final da *string* e usar a estratégia da máquina do item a) para mover a entrada não modificada uma posição para a direita.
4. Se não há mais a's ou b's na fita, reverter os símbolos X e Y para a e b, respectivamente, e parar na posição 0.
5. Caso contrário, avançar até o próximo símbolo original da entrada e voltar ao passo 2.

d. A solução é similar à do item c). Da mesma forma que foi construída uma máquina para inserir um branco entre os símbolos, podemos fazer uma máquina que *remove* um branco entre os símbolos. Assim, o funcionamento geral da máquina pode ser descrito brevemente como a seguir:

1. Procurar o primeiro b na fita a partir da esquerda. Trocar b por B.
2. Mover todo o conteúdo da fita uma posição para a esquerda, eliminando o branco entre os símbolos.
3. Rebobinar até a posição 0 da fita e repetir o passo 2.
4. Parar quando chegar no final da entrada (ler B na busca do passo 1) e rebobinar para o início da fita uma última vez.

3 Construa uma máquina de Turing que computa as funções especificadas nos itens abaixo. (Faça uma máquina para cada item.) Todas as máquinas têm alfabeto de entrada $\Sigma = \{a, b\}$. Os símbolos u e v representam *strings* arbitrárias sobre Σ^*.

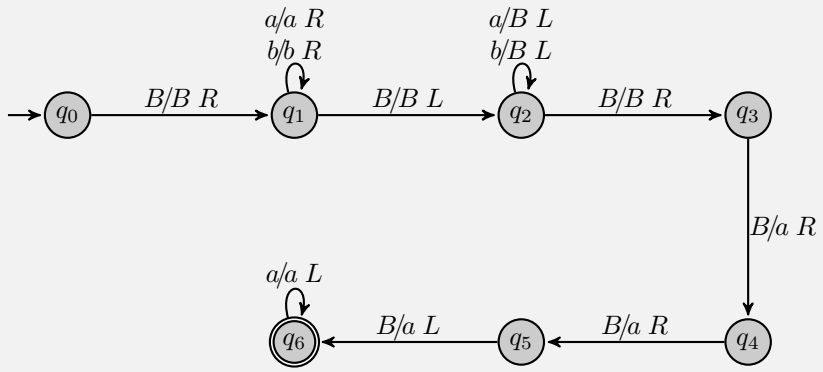
a. $f(u) = aaa$

b. $f(u) = \begin{cases} a & \text{se } \textit{length}\,(u) \text{ é par} \\ b & \text{caso contrário} \end{cases}$

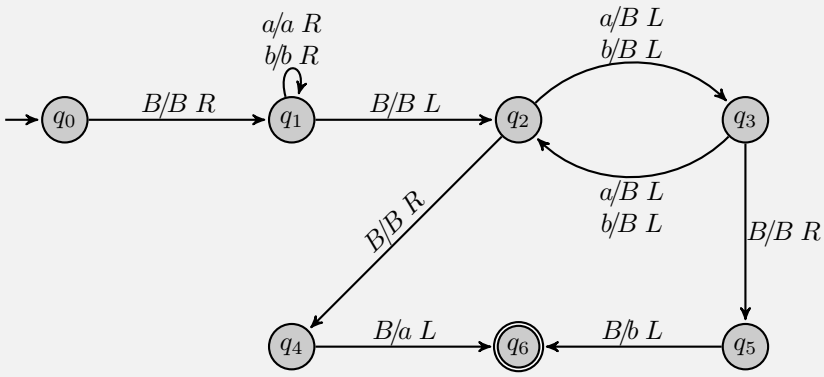
c. $f(u) = u^R$

d. $f(u, v) = \begin{cases} u & \text{se } \textit{length}\,(u) > \textit{length}\,(v) \\ v & \text{caso contrário} \end{cases}$

a. A máquina abaixo apaga a entrada e deixa aaa na fita.



b. A máquina abaixo usa os estados q_2 e q_3 para determinar o tamanho da *string* de entrada. Se a máquina está em q_2, a *string* vista até agora tem tamanho par (assume-se, como qualquer pessoa sana, que 0 é par), e se a máquina está em q_3, o tamanho é impar.



c. A solução é muito similar à máquina da questão 2(b), com uma modificação necessária no estado q_7, aonde ao invés de reverter X e Y para a e b, basta escrever branco. Assim, para uma entrada q_0BuB, a máquina "para" com a configuração $q_7\#B \dots BBu^RB$. Neste ponto, basta mover u^R para a esquerda até obtermos q_fBu^RB. (Note que o branco da posição 0 foi marcado com $\#$ para podermos diferenciar a posição 0 dos demais brancos.)

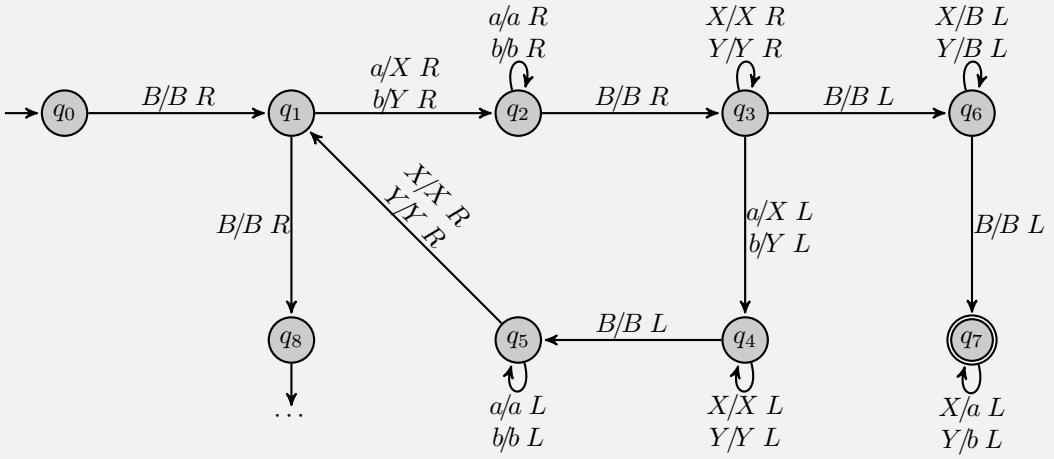
d. A função possui dois argumentos, logo eles são separados por um único espaço na configuração inicial da máquina: q_0BuBvB. A máquina faz um zig-zag entre os dois argumentos, marcando um símbolo de u e um símbolo de v de cada vez. (Essa marcação é feita da forma usual, trocando a por X e b por Y.) Se o zig-zag falhar com a máquina caminhando para direita, isso significa que *length* $(u) >$ *length* (v) e entramos no estado q_6, que apaga v, parando finalmente em q_7 quando u tiver sido revertida. No caso contrário, o zig-zag falha com a máquina caminhando para a esquerda (isto é, buscando um novo símbolo de u para marcar) e temos que *length* $(u) \leq$ *length* (v). Nesse caso, quando detectamos o branco que separa u e v em q_1, a máquina entra em q_8, onde a saída v deve ser preparada. Nesse ponto, a partir de q_8, a máquina realiza as seguintes ações (não desenhadas no diagrama):

   (a) Caminha para a direita revertendo X e Y de v.

   (b) Rebobina a cabeça até o branco separador de u e v.

   (c) Caminha para a esquerda apagando os X's e Y's de u.

   (d) Ao encontrar o branco da posição zero, executa a máquina que move a entrada para a esquerda, realizando a computação $q_iBB \dots BBvB \vdash^* q_fBvB$.



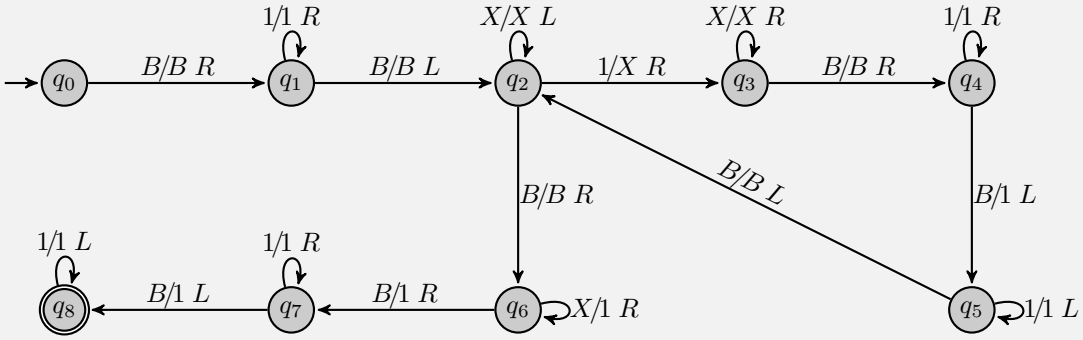
4 Construa uma máquina de Turing que computa as funções numéricas especificadas nos itens abaixo. (Faça uma máquina para cada item.) Não utilize macros nas construções. Utilize a base de representação que preferir.

a. $f(n) = 2n + 3$

b. $eq(n, m) = \begin{cases} 1 & \text{se } n = m \\ 0 & \text{caso contrário} \end{cases}$

a. Vamos usar a notação unária para representar os números naturais, logo, $\Sigma = \{1\}$. Relembrando que um natural $n \in \mathbf{N}$ é escrito em notação unária como $\overline{n} = 1^{n+1}$. Assim, o resultado que deve ficar na fita ao final da computação é $\overline{2n+3} = 1^{2n+4}$. A máquina abaixo copia $\overline{n}$, deixando $2(n+1) = 2n+2$ símbolos 1's na fita. Para obtermos a quantidade necessária de 1's, basta adicionar mais dois 1's para chegar a $2n+4$. Os passos da computação podem ser descritos como abaixo.

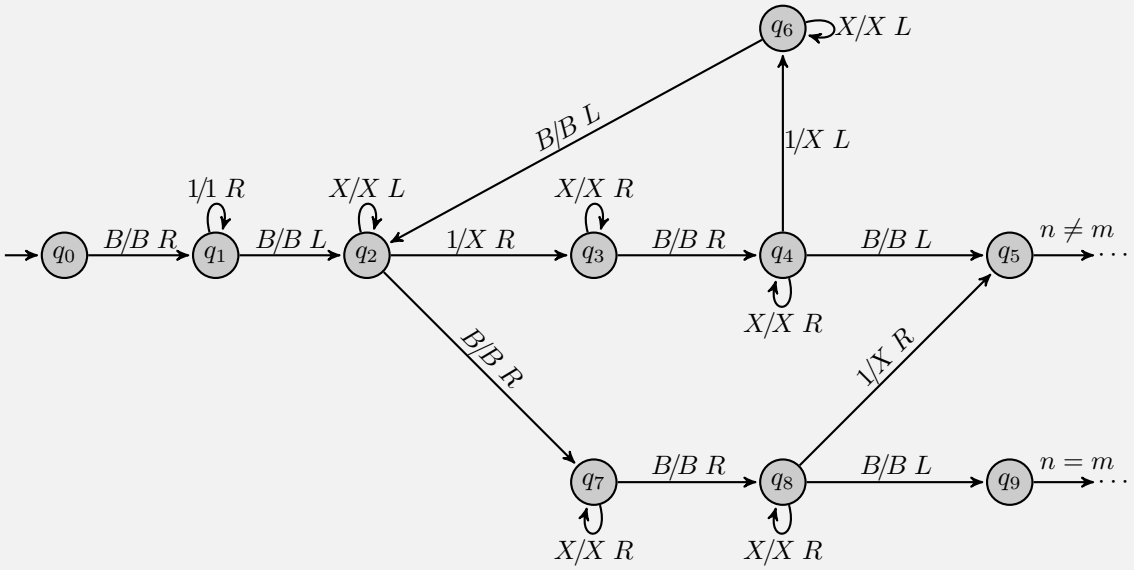
$$B\overline{n}B \vdash^* B\overline{n}B\overline{n}B \vdash^* B\overline{n}1\overline{n}1B = B\overline{2n+3}B$$



b. Como em vários dos exercícios anteriores, para determinar se $n = m$, a máquina faz um zig-zag sobre os 1's de n e m, marcando os símbolos correspondentes com X até que seja possível determinar se $n = m$ ou não. Vale notar que a máquina marca n da direita para a esquerda, e m da esquerda para a direita. Assim, uma possível computação da TM fica como abaixo.

$$B11B11B \vDash B1XB11B \vDash^* B1XBX1B \vDash^* BXXBX1B \vDash^* BXXBXXB \vDash^* B\overline{1}B$$

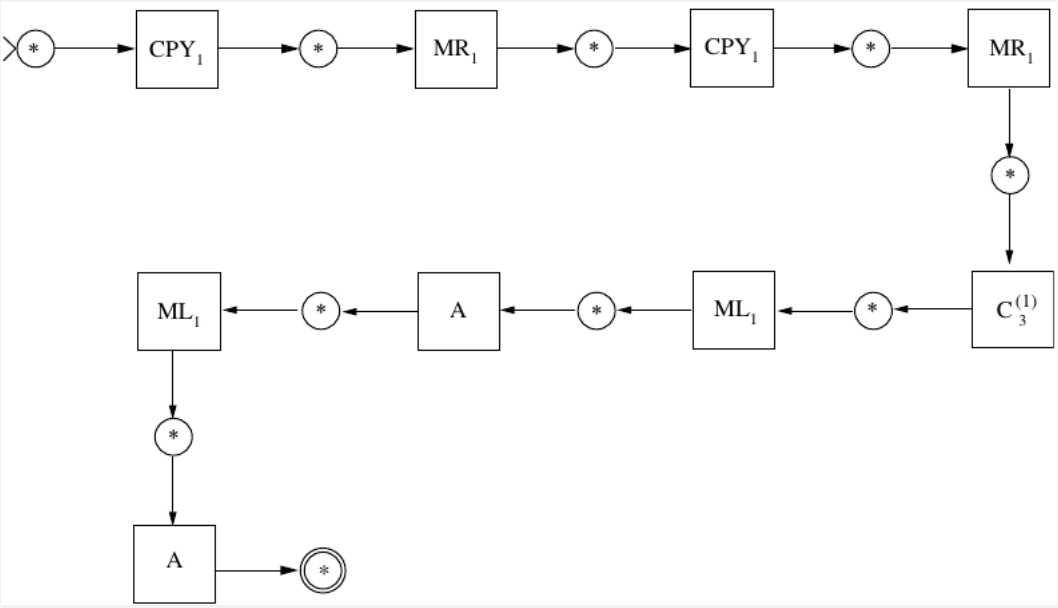
Após encontrar o branco do meio (separador de n e m) no estado q_1, a máquina busca em q_2 o símbolo 1 mais à direita de n. Após marcar essa posição com X, a máquina inicia uma busca por um 1 correspondente em m, caminhando para a direita nos estados q_3 e q_4. Caso a máquina encontre um branco, significa que $n > m$ e a máquina entra em q_5. A partir deste estado, a máquina apaga toda a fita e escreve 0, parando com a configuração $B\overline{0}B$. (Essa parte da máquina não é mostrada na figura.) Caso a máquina consiga marcar um 1 correspondente em m, a máquina caminha para a esquerda procurando o próximo dígito de n e o ciclo se repete. O *loop* termina quando a máquina lê o branco da posição 0 em q_2, entrando em q_7. Nesse estado, sabemos que $n \leq m$ mas ainda é necessário distinguir os casos $n = m$ e $n < m$. Para tal, a máquina caminha até o final da entrada, buscando algum 1 que tenha "sobrado" de m. Se for possível achar um 1, temos que $n < m$ e a máquina vai para q_5 para produzir a saída $B\overline{0}B$. Caso contrário, a máquina entra em q_9, a partir do qual toda a fita é apagada e a computação termina com $B\overline{1}B$ na fita. (Essa parte da máquina também não é mostrada na figura.)



Para garantir o seu entendimento do funcionamento da máquina, realize a computação para as entradas $B11B1B$ e $B1B11B$. Em ambos os casos, a configuração final deve ser $B\overline{0}B$.

5 Use as macros e máquinas definidas entre as seções 9.2 e 9.4 do livro do Sudkamp para projetar uma máquina que computa a função $f(n) = 2n + 3$.

A máquina M que computa a função $f(n) = 2n + 3$ pode ser obtida pela combinação das macros de cópia, soma e funções constantes, como ilustrado abaixo.



A macro $C_3^{(1)}$ computa a função $c(n) = 3$, isto é, a função que sempre retorna a constante 3 para qualquer argumento $n \in \mathbf{N}$. Realizando o *trace* da computação de M para a entrada n, podemos ilustrar o comportamento e a interação das macros que compõem a máquina.

Machine	Configuration
	$\underline{B}\overline{n}B$
CPY_1	$\underline{B}\overline{n}B\overline{n}B$
MR_1	$B\overline{n}\underline{B}\overline{n}B$
CPY_1	$B\overline{n}\underline{B}\overline{n}B\overline{n}B$
MR_1	$B\overline{n}B\overline{n}\underline{B}\overline{n}B$
$C_3^{(1)}$	$B\overline{n}B\overline{n}\underline{B}\overline{3}B$
ML_1	$B\overline{n}\underline{B}\overline{n}B\overline{3}B$
A	$B\overline{n}\underline{B}\overline{n+3}B$
ML_1	$\underline{B}n\underline{B}\overline{n+3}B$
A	$\underline{B}\overline{2n+3}B$

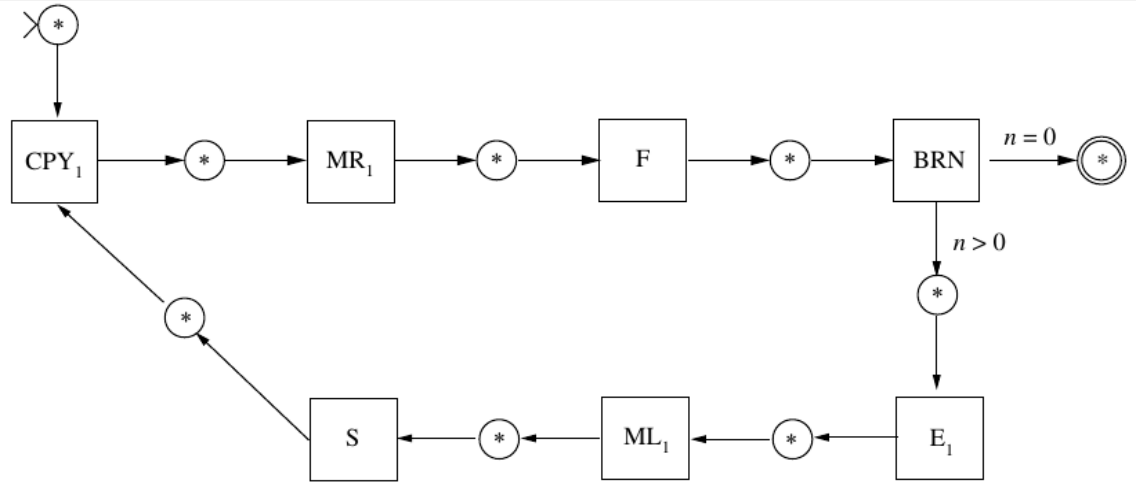
6 Seja F uma máquina de Turing que computa uma função numérica unária e total f. Projete uma máquina M que retorna o primeiro número natural n tal que $f(n) = 0$. A computação de M deve continuar indefinidamente se tal n não existe. Responda os itens abaixo.

a. Apresente M e explique o seu funcionamento.

b. O que aconteceria com a execução de M se a função computada por F não fosse total?

Obs.: Para construir M *você pode utilizar qualquer máquina ou macro vista até aqui.*

a. A computação de M consiste de um ciclo que produz $B\overline{n}B\overline{f(n)}B$, para $n = 0, 1, \ldots$, até que seja encontrado um valor de n para o qual $f(n) = 0$.



A computação começa com o contador $\overline{0}$ na fita. Uma cópia de $\overline{0}$ é feita e F é executada na cópia. A macro BRN é usada para determinar se o valor computado de f é 0. Se não for, o valor computado é apagado, o contador é incrementado e o valor subsequente de f é computado. O ciclo de geração de números naturais e computação do valor de f termina quando for encontrado um n para o qual $f(n) = 0$.

b. Se f nunca assume o valor 0, a computação continua indefinidamente. Se f não é total, a computação vai retornar o primeiro valor n para o qual $f(n) = 0$, mas somente se f for definida para todo $m < n$. Caso contrário, M entra em *loop* ao encontrar o primeiro valor onde $f(m) \uparrow$.

7 Seja F uma máquina de Turing que computa a função numérica unária e total f. Projete uma máquina G que computa a função

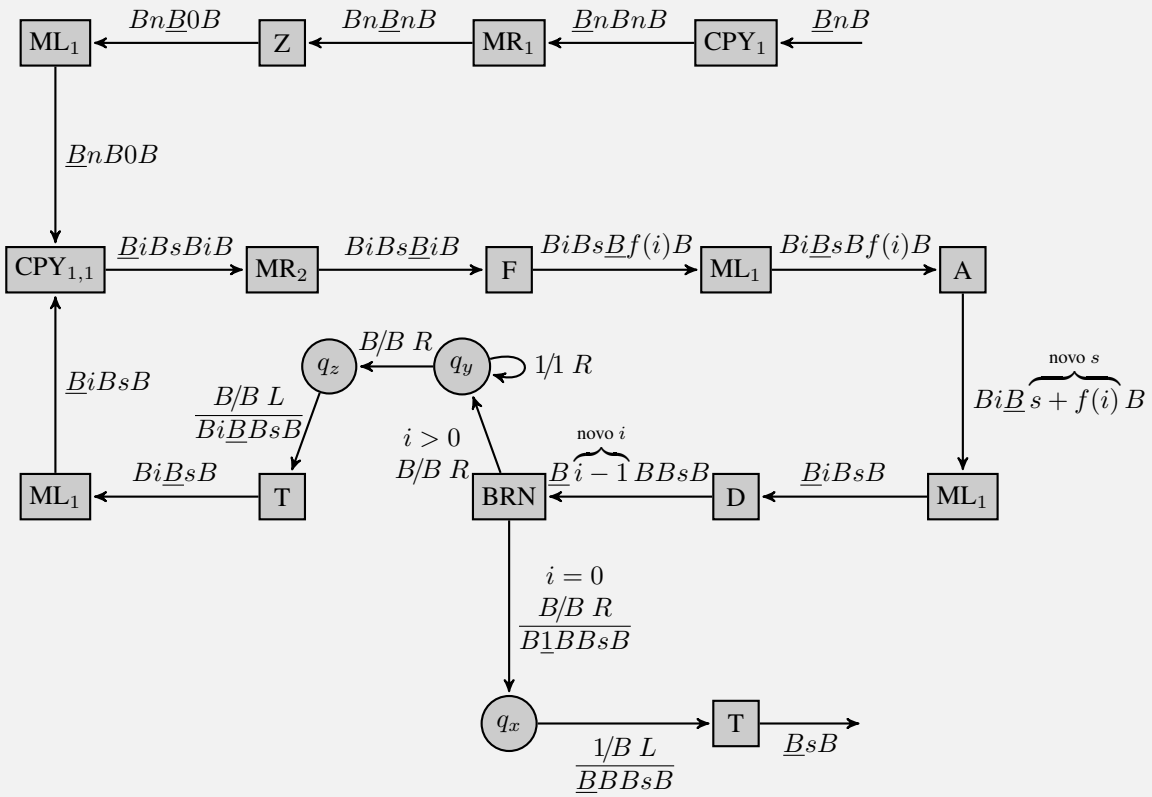
$$g(n) = \sum_{i=1}^{n} f(i) \quad .$$

Obs.: Para construir G *você pode utilizar qualquer máquina ou macro vista até aqui.*

Vamos usar várias macros vistas ao longo do curso. Um ponto fundamental para projetar essa máquina é definir como a fita fica dividida para guardar as informações necessárias para a computação. Como precisamos fazer um somatório, é claro que o projeto da máquina precisa incluir um *loop*. Assim, precisamos armazenar o valor atual da variável i, além do valor s do somatório calculado até o momento. Vamos então usar a fita da seguinte forma: $B\overline{i}B\overline{s}B\overline{f(i)}B$. Em alguns passos da computação, o terceiro campo não estará presente. Além disso, como toda a notação é unária, não vamos mais denotar as variáveis por $\overline{i}$ no diagrama abaixo, escrevendo somente i.

Um ponto que não é essencial mas facilita a construção da máquina é perceber que a soma é uma operação associativa. Assim, não faz diferença calcular o somatório começando com $i = 1$ e indo até $i = n$, podemos usar o contador no sentido inverso. Assim, começando com $i = n$, decrementamos i até zero para usarmos a macro BRN.

A máquina completa é mostrada na figura abaixo. Destacamos o formato da entrada e saída de cada macro para garantir consistência da composição das macros.



8 Sejam F e G máquinas de Turing que computam, respectivamente, as funções numéricas unárias e totais f e g. Projete uma máquina H que computa a função

$$h(n) = \sum_{i=1}^{n} eq(f(i), g(i)) \quad .$$

Isto é, $h(n)$ é a quantidade de valores entre 1 e n para os quais as funções f e g assumem o mesmo valor.
Obs.: Para construir H *você pode utilizar qualquer máquina ou macro vista até aqui.*

A solução é muito parecida com a do exercício anterior. Não vamos apresentar a máquina completa mas destacar os passos principais do seu funcionamento. O ponto fundamental de divisão da fita continua o mesmo. Vamos usar a divisão a seguir: $BiBsBf(i)Bg(i)B$. Inicialmente a fita começa com BnB. Como no exercício anterior, esse valor é copiado e entregue para a máquina Z, deixando $BnB0B$ na fita. A seguir entramos no seguinte *loop*:

1. Partindo de $BiBsB$, copiar i depois de s, gerando $BiBsBiB$.
2. Executar F sobre o i recém copiado, obtendo $BiBsBf(i)B$.
3. Voltar ao início da fita e copiar i depois s e $f(i)$, chegando a $BiBsBf(i)BiB$.
4. Executar G sobre o i recém copiado, obtendo $BiBsBf(i)Bg(i)B$.
5. Executar EQ sobre $f(i)$ e $g(i)$, gerando $BiBsBeq(f(i), g(i))B$.
6. Acumular em s o seu valor atual com o resultado de EQ.
7. Decrementar i e testar o seu valor. Se $i > 0$, retorne ao passo 1. Caso contrário, ajuste o conteúdo da fita para deixar como resultado final BsB.