Aula 01 – Máquinas de Turing e Funções Computáveis

Prof. Eduardo Zambon

Departamento de Informática (DI)
Centro Tecnológico (CT)
Universidade Federal do Espírito Santo (Ufes)

Algoritmos e Fundamentos da Teoria de Computação (ToCE)
Engenharia de Computação

Introdução

- O nosso mecanismo de computação abstrata é a Máquina de Turing (TM).
- Uma TM serve para computar uma função.
- $\Rightarrow$ Aproxima a abstração da TM de um computador real.
- **Estes *slides***: definição, conceitos e propriedades de funções computáveis.
- **Objetivos**: apresentar TMs que realizam computações numéricas.

Referências

Section 8.1 & Chapter 9 – Turing Computable Functions
T. Sudkamp
Section 3.1 – Turing Machines
M. Sipser
Section 4.1 – Definition of a Turing Machine
A. Maheshwari

Introdução

- Computabilidade: estudo das capacidades e limitações da computação algorítmica.
- Procedimento efetivo (PE): processos intuitivamente considerados computáveis.
- Um procedimento efetivo consiste de:
  - Uma entrada a ser processada.
  - Um conjunto de instruções.
  - Uma especificação da ordem de execução das instruções.
- Execução de uma instrução é mecânica: não requer entendimento, inteligência ou ingenuidade do executor.
- Computação (efetiva) produzida por um PE executa um número finito de instruções e termina.
- Em suma: um PE é um processo discreto e determinístico que termina para todas as possíveis entradas.

Máquinas de Turing

- A máquina de Turing (TM) é uma evolução de autômatos finitos.
- Proposta por Alan Turing em 1936.
- Objetivo original: modelo abstrato para o estudo de computações efetivas.
- *By design*: TM serviu de modelo para o projeto e desenvolvimento do *stored-program computer (Von Neumann/Harvard arch.)*.
- Operações básicas permitem ler/escrever qualquer posição de memória.
- Diferença para um computador real: uma TM não possui limitação quanto à quantidade de tempo/memória disponível para uma computação.

A Máquina de Turing Padrão

- Estrutura de uma TM é similar ao de um autômato finito.
- Características adicionais incorporadas na função de transição.
- Uma TM é um autômato finito aonde uma transição escreve um símbolo na fita.
- A cabeça da fita pode se mover em ambas direções.
- Permite que a máquina manipule a entrada várias vezes.

A Máquina de Turing Padrão

Definição 8.1.1 (Sudkamp)

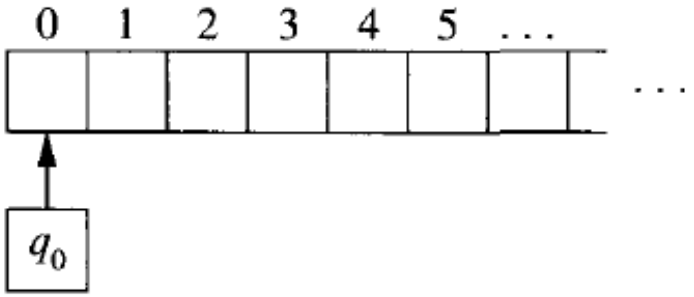
Uma máquina de Turing é uma tupla $M = (Q, \Sigma, \Gamma, \delta, q_0)$, onde:

- Q é um conjunto finito de estados.
- Γ é o alfabeto da fita, que contém o símbolo especial B representando um branco.
- $\Sigma \subset \Gamma - \{B\}$ é o alfabeto de entrada.
- $\delta : Q \times \Gamma \rightharpoonup Q \times \Gamma \times \{L, R\}$ é a função de transição.
- $q_0 \in Q$ é o estado inicial.

Os símbolos L, R indicam o movimento da cabeça da fita uma posição à esquerda ou à direita, respectivamente.

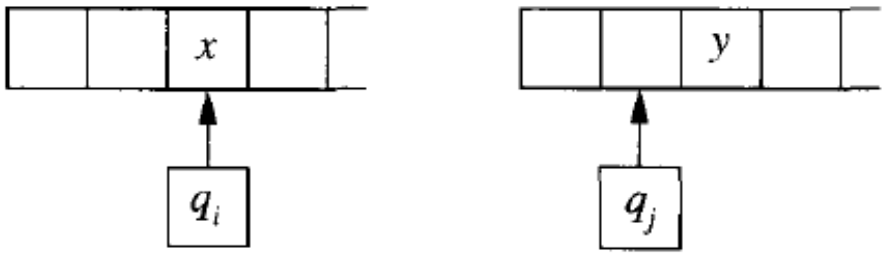
A Máquina de Turing Padrão

- A fita da TM tem um limite à esquerda e se estende indefinidamente à direita.
- As posições da fita são numeradas pelos naturais, começando do zero.
- Cada posição da fita contém um elemento de Γ.
- Computação da TM começa no estado q_0 com a cabeça da fita na posição 0.
- A string de entrada $s \in \Sigma^*$ começa na posição 1.
- A posição 0 e as demais além da entrada estão em branco (símbolo B).



A Máquina de Turing Padrão

- Símbolos em $\Gamma - \Sigma$ provêm representações adicionais que podem ser usadas durante uma computação.
- Uma transição causa três ações:
  1. Mudar de um estado para outro.
  2. Escrever um símbolo na posição acessada pela cabeça da fita.
  3. Mover a cabeça da fita.
- A direção do movimento é dada pelo último componente da transição.
- Resultado da transição $\delta(q_i, x) = [q_j, y, L]$:



A Máquina de Turing Padrão

- Possibilidade de mover em ambas as direções e processar brancos $\Rightarrow$ computação pode continuar indefinidamente.
- Uma TM trava quando encontra um par de estado e símbolo (não-branco) sem uma transição definida.
- Se isso ocorre dizemos que a computação terminou anormalmente.
- Computação com terminação normal: cabeça da fita na posição 0 lendo B.
- TM da Definição 8.1.1 é determinística.
- Máquina de Turing Padrão: TM determinística com uma fita (como acima) e uma cabeça de fita.
- Existem variações de TMs com mais fitas e comportamento não-determinístico: próximas aulas.

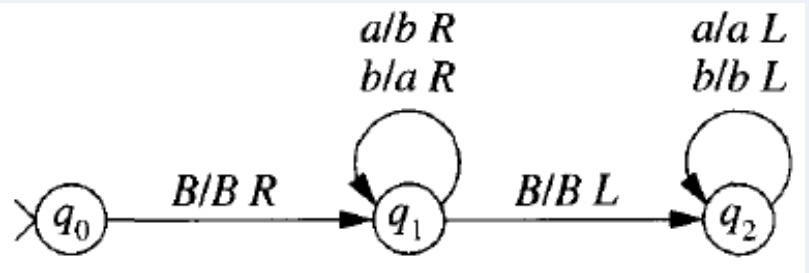
A Máquina de Turing Padrão

Exemplo 8.1.1 (Sudkamp)

A função de transição de uma TM padrão com alfabeto $\Sigma = \{a, b\}$ é dada na tabela abaixo.

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

A TM pode ser representada pelo diagrama de estados abaixo.



A Máquina de Turing Padrão

- Configuração de uma TM: uq_ivB.
- uv é a string atualmente na fita começando na posição 1.
- Todas as posições à direita de uvB são branco.
- Note que podem haver brancos em uv.
- O estado atual da máquina é q_i e a cabeça da fita está acessando o primeiro símbolo de v.
- Configurações da máquina podem ser usadas para fazer *traces* da computação.
- $uq_ivB \vdash xq_jyB$: configuração à direita é obtida da configuração à esquerda através de uma única transição da TM.
- $uq_ivB \vdash^* xq_jyB$: zero ou mais transições.

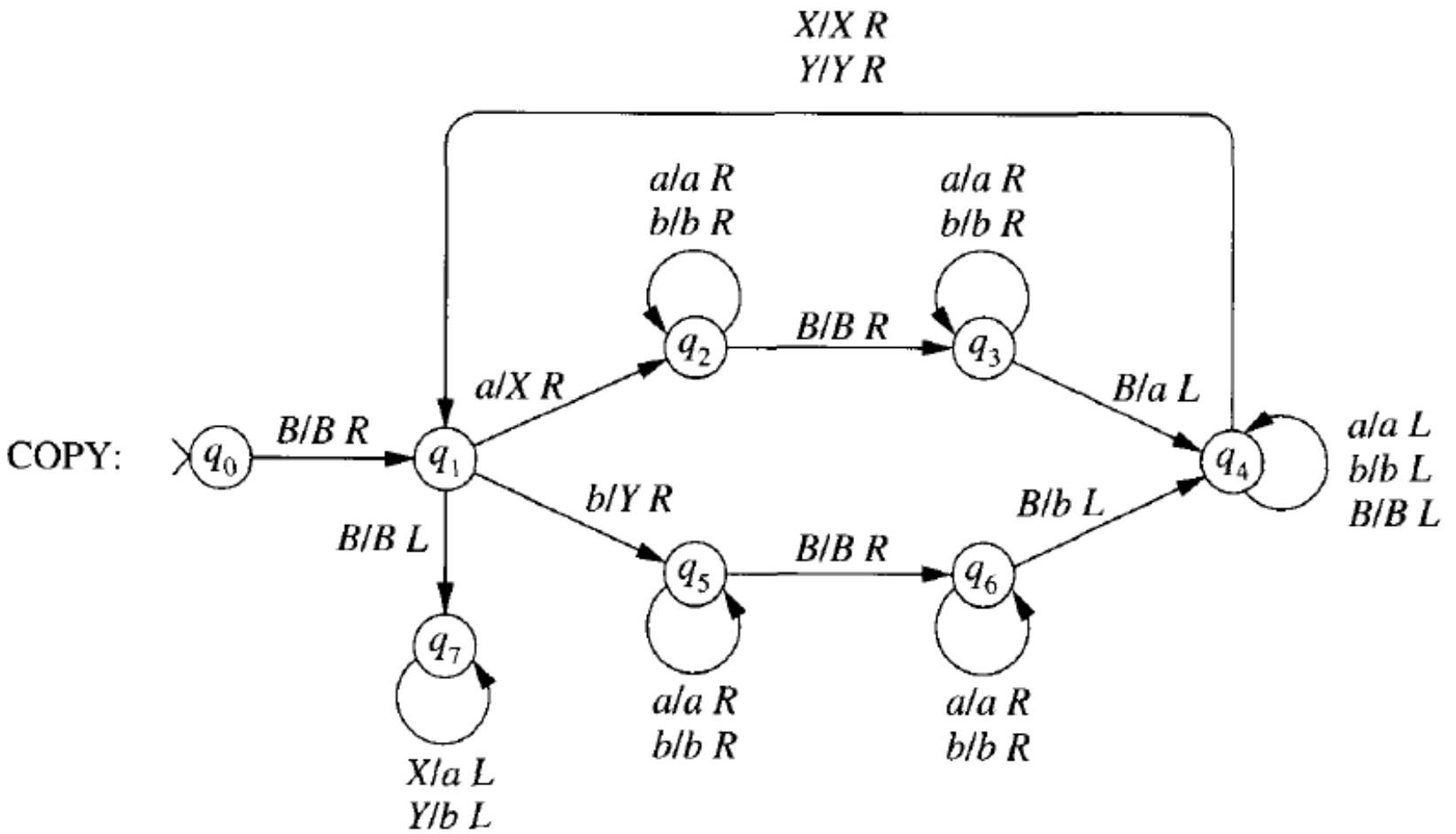
A Máquina de Turing Padrão

Trace da computação da máquina do Exemplo 8.1.1 para a string de entrada *abab*.

$$
\begin{aligned}
& q_0 B a b a b B \\
\vdash\ & B q_1 a b a b B \\
\vdash\ & B b q_1 b a b B \\
\vdash\ & B b a q_1 a b B \\
\vdash\ & B b a b q_1 b B \\
\vdash\ & B b a b a q_1 B \\
\vdash\ & B b a b q_2 a B \\
\vdash\ & B b a q_2 b a B \\
\vdash\ & B b q_2 a b a B \\
\vdash\ & B q_2 b a b a B \\
\vdash\ & q_2 B b a b a B .
\end{aligned}
$$

Exemplo 8.1.2 (Sudkamp)

A TM COPY com $\Sigma = \{a, b\}$ produz uma cópia da entrada.



Computando Funções – Introdução

- Computações de uma TM produzem um mapeamento entre strings de entrada e saída.
- $\Rightarrow$ A TM computa uma função.
- Strings de entrada e saída podem ser interpretadas como números naturais.
- Nessa caso, TMs computam funções numéricas (*number-theoretic functions*).

Computando Funções – Revisão

Relembrando:

- Uma função (total) $f : X \to Y$ é um mapeamento que associa no máximo um valor em Y para cada $x \in X$.
- $f(x)\uparrow$ indica que f não é definida para x.
- $f(x)\downarrow$ indica que f é definida mas o valor não importa.
- Se $f(x)\uparrow$ para algum $x \in X$, então f é uma função parcial, denotada como $f : X \rightharpoonup Y$.

- Definição de uma função f não especifica como obter (computar) $f(x)$ para uma entrada x.
- Definição matemática de f em geral é declarativa e não construtiva.
- Uma TM que computa f pode ser vista como uma definição construtiva de f.

Computando Funções

- O domínio e co-domínio de uma função computada por uma TM são ambos Σ^*.
- As strings de Σ^* podem representar números em alguma base.
- Uma TM que computa uma função tem exatamente um estado inicial q_0 e um estado final q_f.
- A entrada é posicionada como anteriormente (TM padrão).
- Computação sempre começa com uma transição de q_0 que posiciona a cabeça no começo da string de entrada.
- TM nunca retorna ao estado inicial.
- Todas as computações que terminam, o fazem no estado q_f, com o valor da função (saída) na mesma posição da entrada.
- Condições formalizadas como a seguir.

Computando Funções

Definição 9.1.1 (Sudkamp)

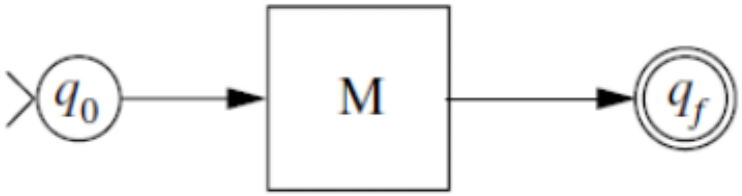
Uma máquina de Turing $M = (Q, \Sigma, \Gamma, \delta, q_0, q_f)$ computa a função $f : \Sigma^* \to \Sigma^*$ se:

1. $\delta(q_0, B) = [q_i, B, R]$ é a única transição definida para q_0.
2. Não há nenhuma transição chegando em q_0.
3. Não há nenhuma transição da forma $\delta(q_f, B)$.
4. A TM M para e realiza a computação $q_0BuB \vdash^* q_fBvB$ sempre que $f(u) = v$.
5. A TM M não para quando $f(x)\uparrow$.

- Uma função f é dita computável (*Turing computable*) se existe uma TM que computa f.
- TMs podem computar tanto funções totais quanto parciais.

Computando Funções

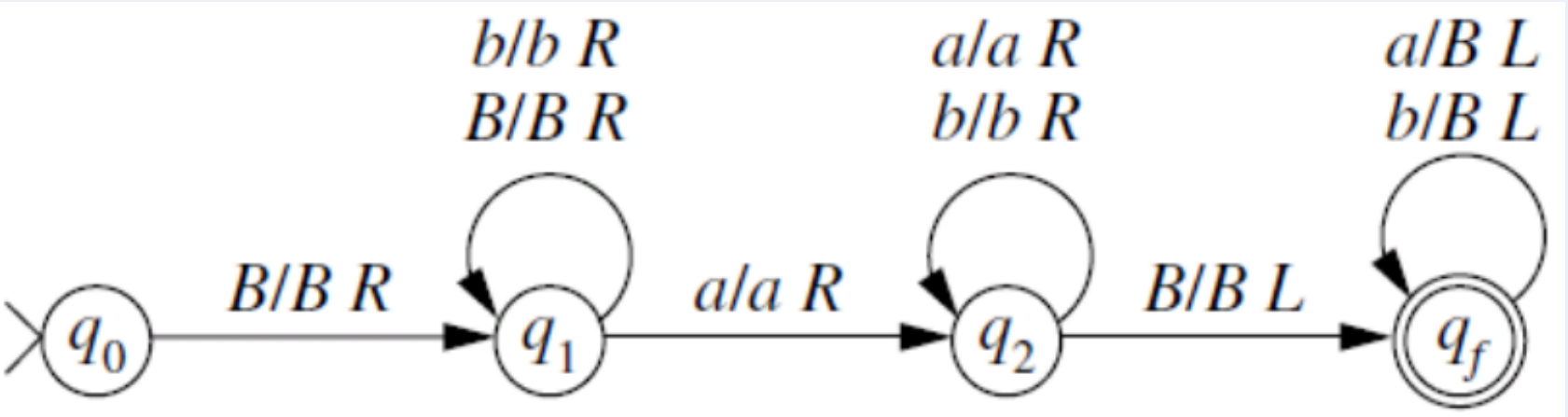
Uma máquina M que computa uma função pode ser representada como ao lado.



Exemplo 9.1.1 (Sudkamp)

A TM abaixo computa a função parcial $f : \{a, b\}^* \rightharpoonup \{a, b\}^*$ defina como

$$f(u) = \begin{cases} \lambda & \text{se } u \text{ contém um } a \\ \uparrow & \text{caso contrário.} \end{cases}$$



$\Rightarrow$ O que acontece se a entrada for *BbBaB*?

Computando Funções

Exercício

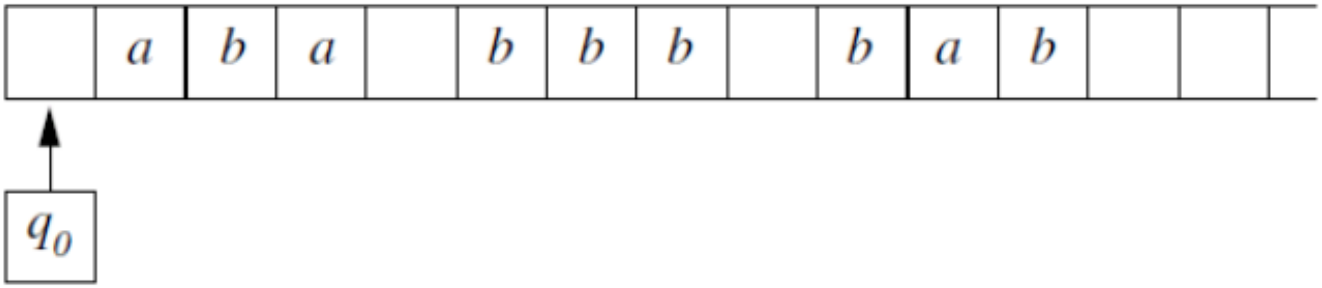
Construa uma TM que computa a função

$$f(n) = \begin{cases} n/2 & \text{se } n \text{ é um múltiplo de 2} \\ \uparrow & \text{caso contrário.} \end{cases}$$

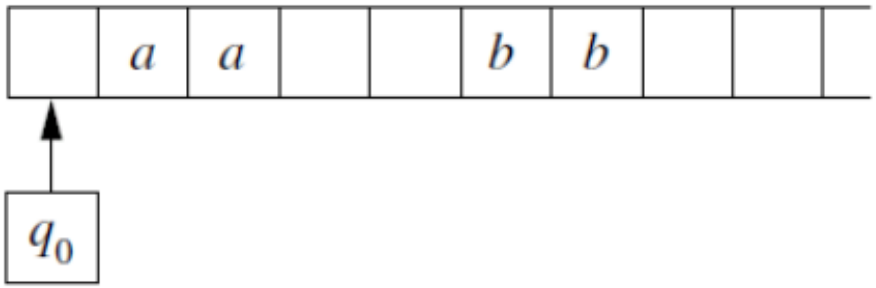
Considere que as strings de entrada e saída são números naturais em notação binária.

Computando Funções

- A entrada para funções com mais de um argumento fica separada por brancos.
- Exemplo 1: configuração para a entrada (aba, bbb, bab).

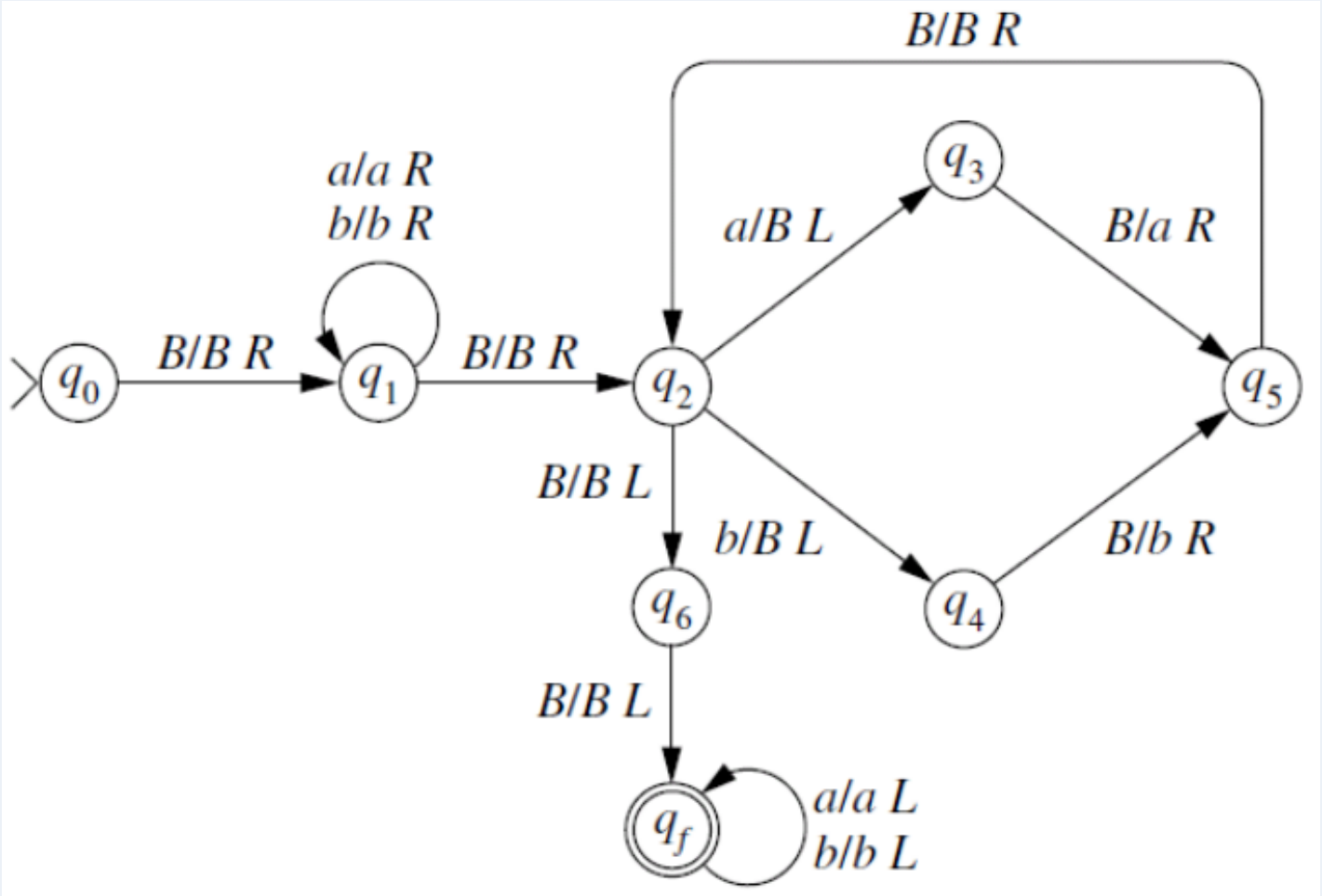


- Exemplo 2: configuração para a entrada (aa, λ, bb).



Exemplo 9.1.2 (Sudkamp)

A TM abaixo computa a função de concatenação de strings.

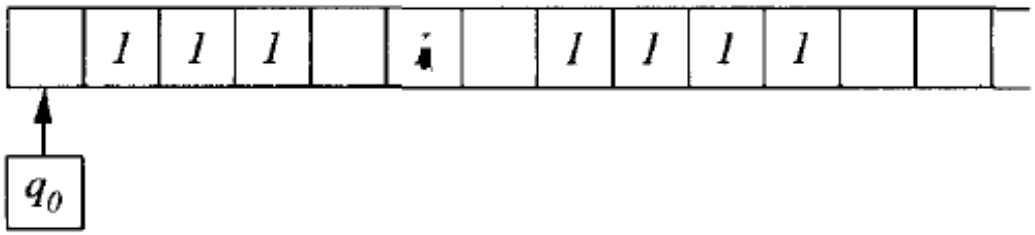


Computação Numérica

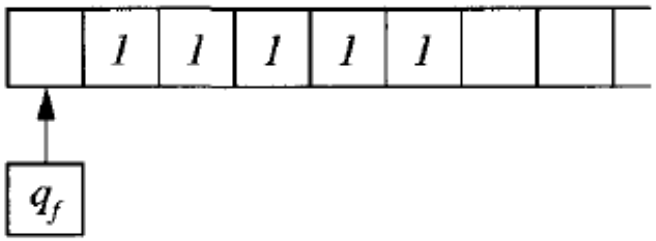
- Uma função numérica é uma função da forma $f : \mathbf{N} \times \mathbf{N} \times \cdots \times \mathbf{N} \to \mathbf{N}$.
- Exemplo: a função $sq : \mathbf{N} \to \mathbf{N}$ definida como $sq(n) = n^2$ é uma função numérica unária.
- Operações padrão de soma e multiplicação são funções numéricas binárias.
- Transição de computação simbólica para numérica requer somente um ajuste de representação.
- Números naturais são representados por strings de símbolos 1.
- O número n é representado pela string 1^{n+1}, isto é, uma string com $n + 1$ "1"s consecutivos.

Computação Numérica

- Exemplos: 0 = “1”, 1 = “11”, 5 = “111111”.
- Essa codificação é chamada de representação únaria dos números naturais.
- Não confundir com função unária: mesmo nome, outro conceito.
- A representação unária de um natural n é escrita como $\overline{n}$.
- TM com representação unária $\Rightarrow \Sigma = \{1\}$.
- Exemplo: entrada para $f(2,0,3)$

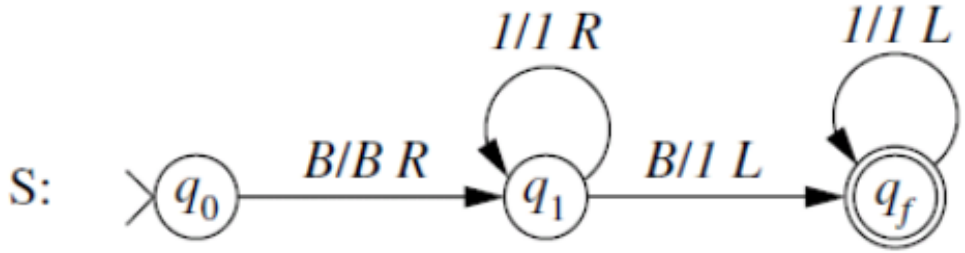


- Saída se $f(2,0,3) = 4$

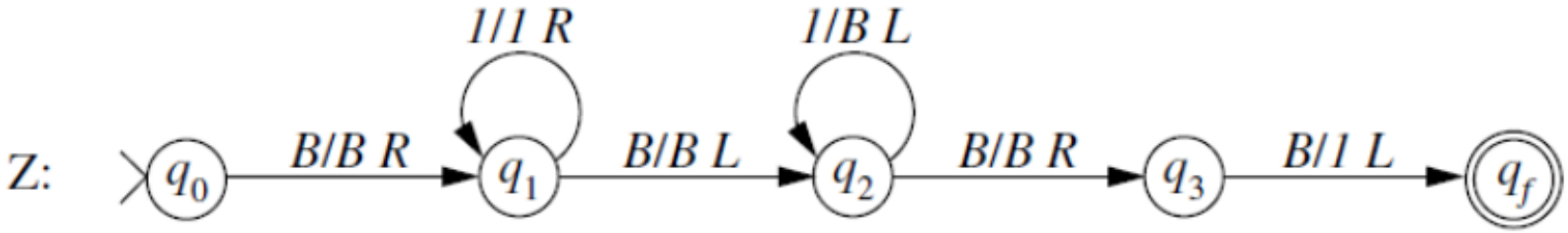


Exemplos de TMs para Funções Numéricas

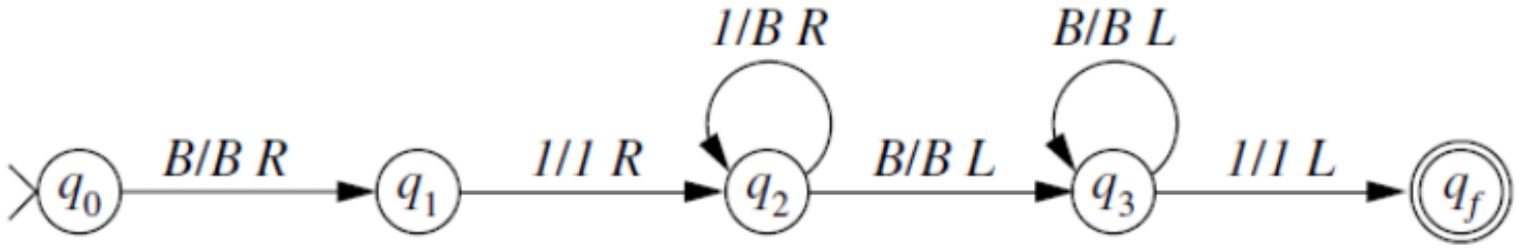
Função sucessor $s(n) = n + 1$



Função zero $z(n) = 0$



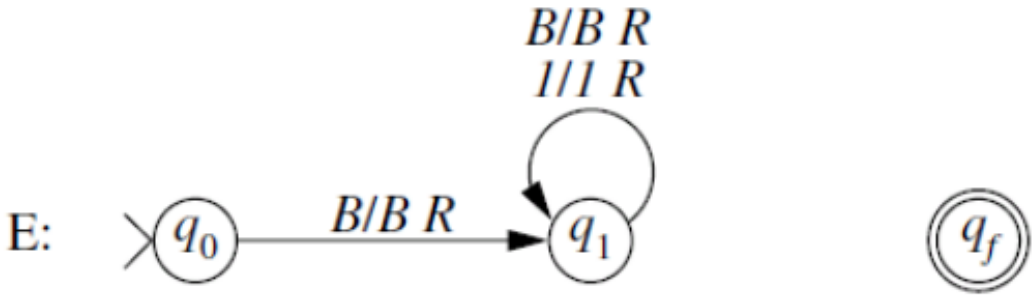
Alternativamente



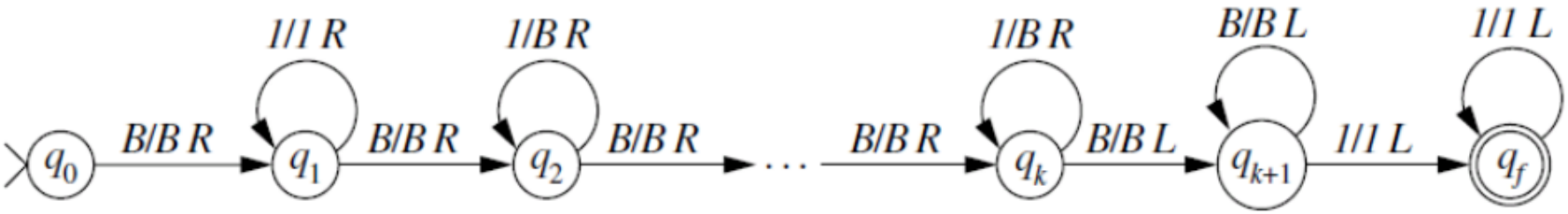
Duas TMs que computam a mesma função $\Rightarrow$ diferença entre definição de função e algoritmo.

Exemplos de TMs para Funções Numéricas

Função vazia: $e(n) = \uparrow$

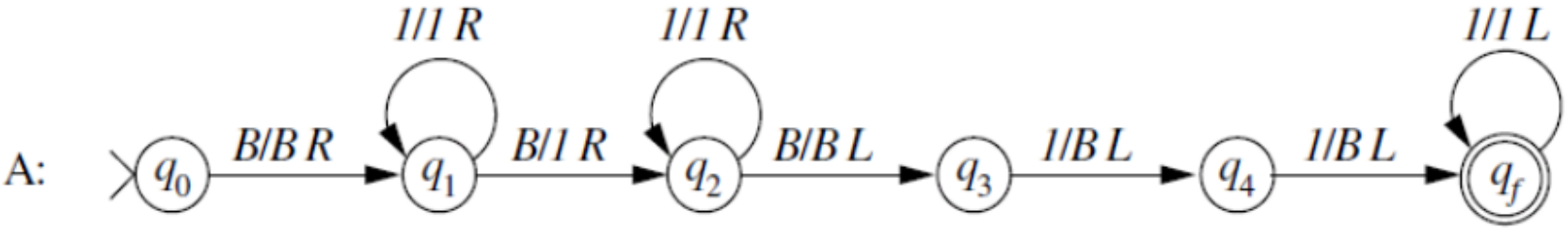


Função projeção $p_i^{(k)}(n_1, \ldots, n_k) = n_i$. Abaixo a TM para $p_1^{(k)}$:



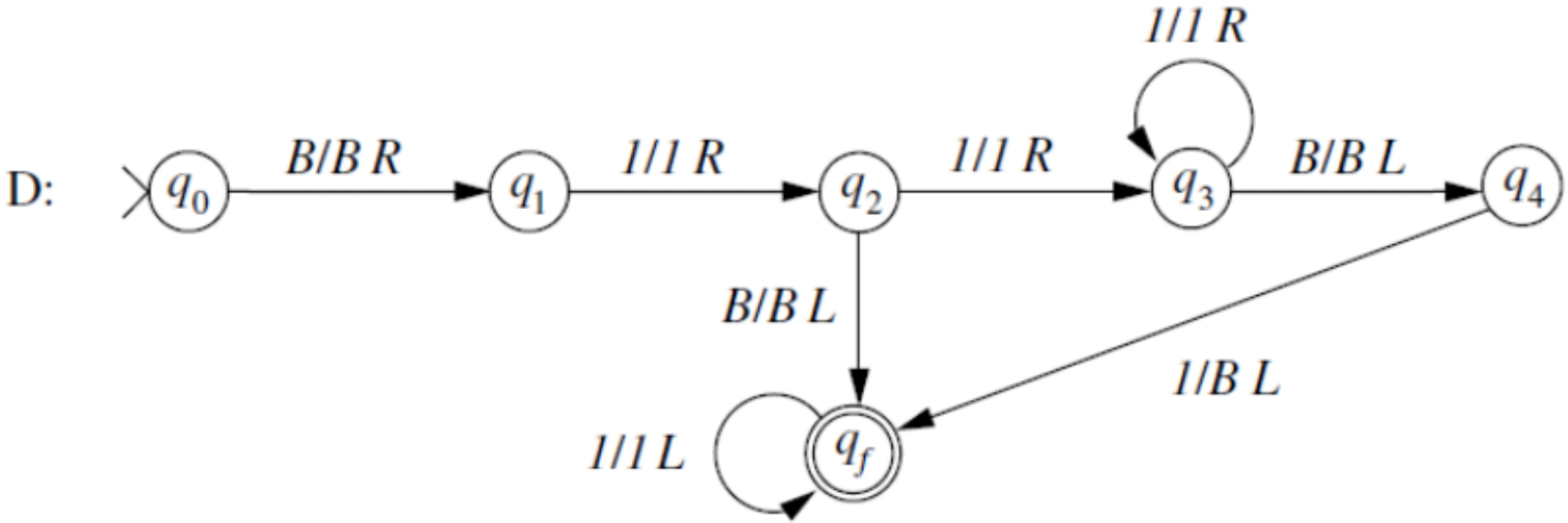
Exemplos de TMs para Funções Numéricas

Função binária de adição: $1^{m+1} + 1^{n+1} = 1^{m+n+1}$.



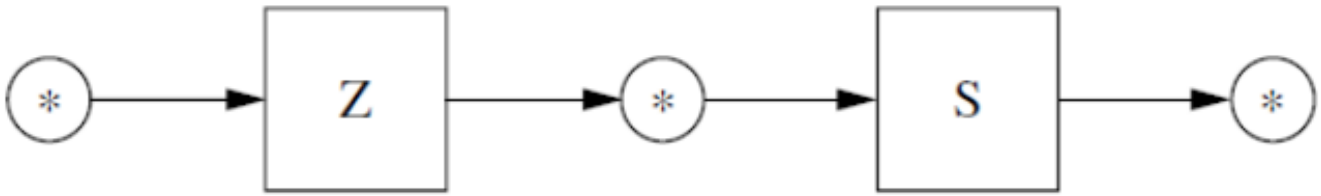
Insere um 1 no meio dos argumentos e apaga dois 1 do final.

Função predecessor $d(0) = 0$, $d(n+1) = n$

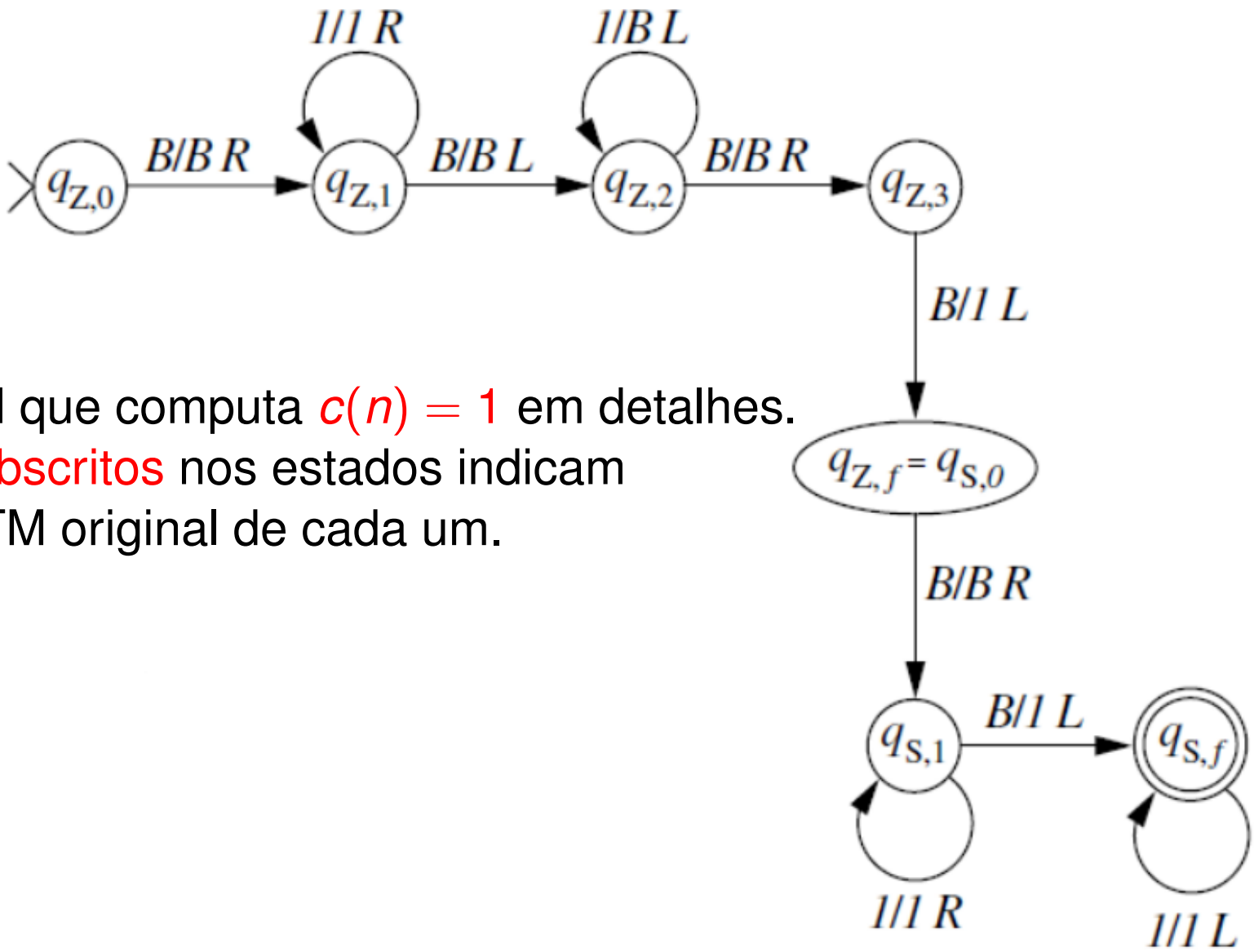


Operação Sequencial de TMs

- TMs projetadas para realizar uma tarefa simples podem ser combinadas para construir TMs que realizam tarefas complexas.
- Combinação: execução sequencial das TMs.
- Resultado de uma computação vira a entrada da TM seguinte.
- Exemplo: uma TM que computa a função constante $c(n) = 1$ pode ser construída a partir da função (TM) zero seguida da TM sucessor.
- Representada pelo diagrama abaixo.



Operação Sequencial de TMs



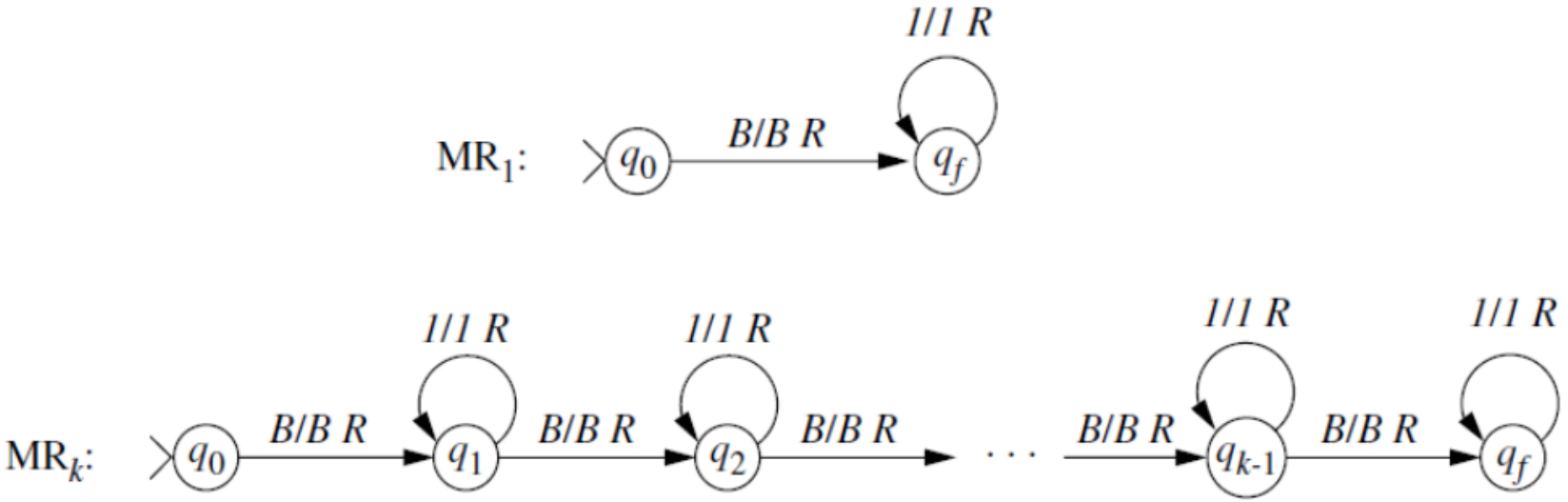
TM que computa $c(n) = 1$ em detalhes.
Subscritos nos estados indicam
a TM original de cada um.

Operação Sequencial de TMs

- Certas ações ocorrem com frequência na computação de TMs.
- $\Rightarrow$ Projetar TMs para realizar essas tarefas recorrentes.
- Projetadas de forma a facilitar o seu uso em máquinas mais complexas.
- TMs desse tipo são chamadas de macros.
- Condições da Definição 9.1.1 são relaxadas um pouco:
  - Computação de uma macro não precisa começar com a cabeça da fita na posição zero.
  - Primeiro símbolo lido deve ser um branco. (Igual)
  - A entrada pode estar imediatamente à esquerda ou a direita da posição inicial.
  - Computação pode terminar em diferentes estados.
  - Não há transições saindo de um estado final. (Igual)

Operação Sequencial de TMs

- Famílias de macros são descritas por esquemas.
- O esquema MR_i (*move right*) move a cabeça para a direita passando por *i* naturais consecutivos em notação unária.



Operação Sequencial de TMs

- O esquema de macros MR não modifica a fita à esquerda da posição inicial da cabeça.
- Computação de MR_2 que começa na configuração de fita

$$\overline{Bn_1}\, q_0\, \overline{Bn_2}\, \overline{Bn_3}\, \overline{Bn_4}$$

termina na configuração

$$\overline{Bn_1}\, \overline{Bn_2}\, \overline{Bn_3}\, q_f\, \overline{Bn_4}.$$

- Macros, assim como TMs, esperam a entrada de uma certa forma.
- O projeto de uma TM composta deve garantir que cada macro receba a configuração de entrada apropriada.

Operação Sequencial de TMs

- Macros podem ser descritas pelo seu efeito sobre a fita.
- Sublinhado: localização da cabeça da fita.
- Setas duplas: correspondência entre mesmas posições.

ML_k (move left):

$$\begin{array}{lll} B\overline{n}_1 B\overline{n}_2 B \ldots B\overline{n}_k \underline{B} & & k \geq 0 \\ \updownarrow \qquad\qquad\qquad \updownarrow & & \\ \underline{B}\overline{n}_1 B\overline{n}_2 B \ldots B\overline{n}_k B & & \end{array}$$

FR (find right):

$$\begin{array}{ll} \underline{B} B^i \overline{n} B & i \geq 0 \\ \updownarrow \quad \updownarrow & \\ B^i \underline{B} \overline{n} B & \end{array}$$

Operação Sequencial de TMs

FL (find left):

$$\begin{array}{ll} B\overline{n}B^{i}\underline{B} & i \geq 0 \\ \updownarrow \quad \updownarrow & \\ \underline{B}\overline{n}B^{i}B & \end{array}$$

E_k (erase):

$$\begin{array}{lll} \underline{B}\overline{n}_1 B\overline{n}_2 B \ldots B\overline{n}_k B & & k \geq 1 \\ \updownarrow \qquad\qquad\qquad\qquad \updownarrow & & \\ \underline{B}B \qquad\qquad \ldots \qquad\qquad BB & & \end{array}$$

Operação Sequencial de TMs

CPY_k (copy):

$$\begin{array}{lll} \underline{B}\overline{n}_1 B \overline{n}_2 B \ldots B \overline{n}_k B B B & \ldots \quad B B & k \geq 1 \\ \updownarrow \qquad\qquad \updownarrow & \updownarrow & \\ \underline{B}\overline{n}_1 B \overline{n}_2 B \ldots B \overline{n}_k B \overline{n}_1 B \overline{n}_2 B \ldots B \overline{n}_k B & & \end{array}$$

$\mathrm{CPY}_{k,i}$ (copy through i numbers):

$$\begin{array}{lll} \underline{B}\overline{n}_1 B \overline{n}_2 B \ldots B \overline{n}_k B \overline{n}_{k+1} \ldots B \overline{n}_{k+i} B B & \ldots \quad B B & k \geq 1 \\ \updownarrow \qquad\qquad \updownarrow \qquad\qquad \updownarrow & \updownarrow & \\ \underline{B}\overline{n}_1 B \overline{n}_2 B \ldots B \overline{n}_k B \overline{n}_{k+1} \ldots B \overline{n}_{k+i} B \overline{n}_1 B \overline{n}_2 B \ldots B \overline{n}_k B & & \end{array}$$

Operação Sequencial de TMs

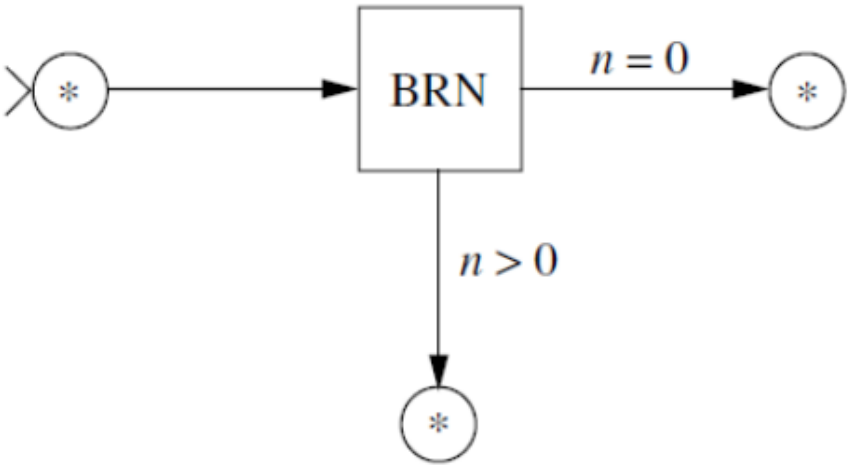
T (translate):

$$\underline{B}B^i\overline{n}B \qquad i \geq 0$$

$$\updownarrow \quad \updownarrow$$

$$\underline{B}\overline{n}B^iB$$

BRN (branch on zero):



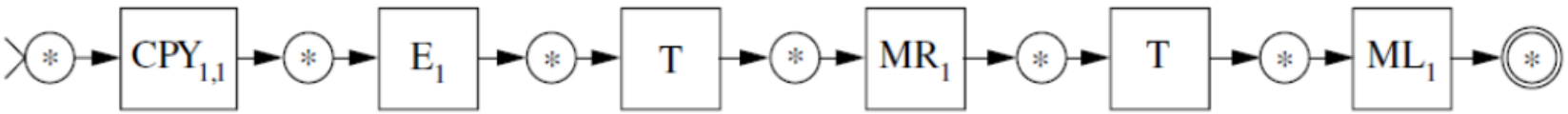
Computando Funções

Exercício

Apresente uma TM para a macro BRN.

Operação Sequencial de TMs

- Macros podem ser compostas como TMs.
- Macro abaixo troca (*interchanges* – INT) a ordem de dois números.



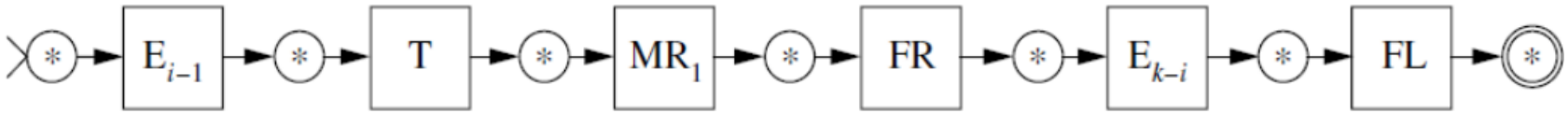
$$\underline{B}\overline{n}B\overline{m}BB^{n+1}B$$

$$\updownarrow \qquad\qquad \updownarrow$$

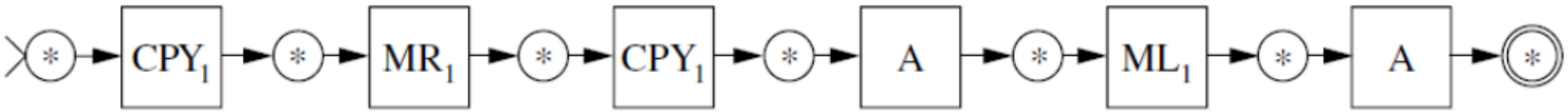
$$\underline{B}\overline{m}B\overline{n}BB^{n+1}B$$

Operação Sequencial de TMs

Função de projeção $p_i^{(k)}$:



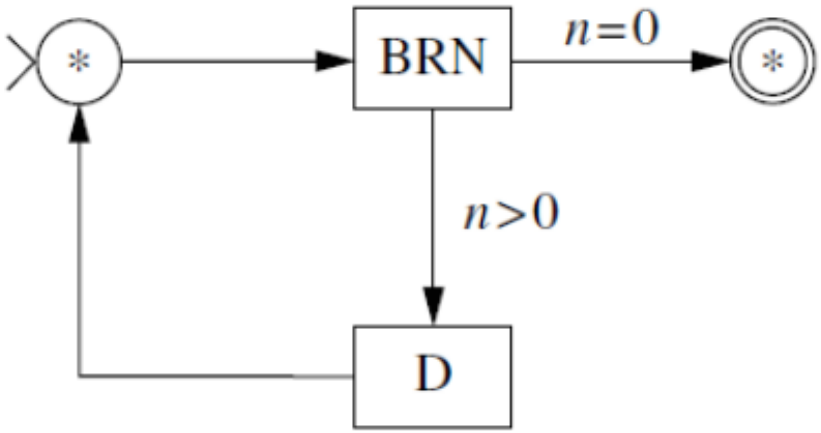
Função $f(n) = 3n$:



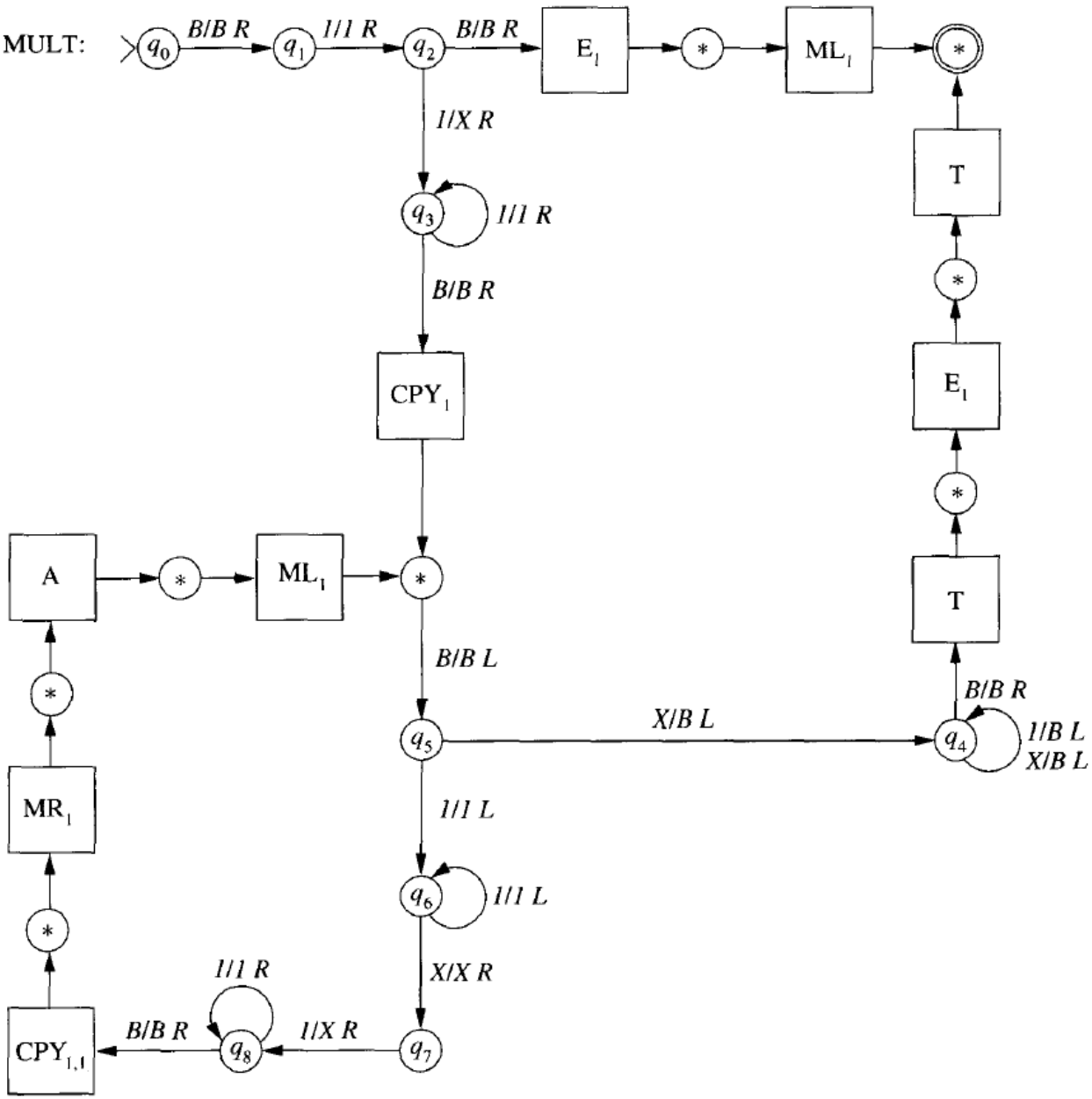
Machine	Configuration
	$\underline{B}\overline{n}B$
CPY_1	$\underline{B}\overline{n}B\overline{n}B$
MR_1	$B\overline{n}\underline{B}\overline{n}B$
CPY_1	$B\overline{n}\underline{B}\overline{n}B\overline{n}B$
A	$B\overline{n}\underline{B}\overline{n+n}B$
ML_1	$\underline{B}nB\overline{n+n}B$
A	$\underline{B}\overline{n+n+n}B$

Operação Sequencial de TMs

Função zero usando a macro BRN e a máquina D (predecessor).



Operação Sequencial de TMs – Multiplicação (MULT)



Composição de Funções

Definição 9.4.1 (Sudkamp)

Sejam g e h duas funções numéricas unárias. A composição de g com h é a função unária $f : \mathbf{N} \to \mathbf{N}$ definida por

$$f(x) = \begin{cases} \uparrow & \text{se } g(x)\uparrow \\ \uparrow & \text{se } g(x) = y \text{ e } h(y)\uparrow \\ h(y) & \text{se } g(x) = y \text{ e } h(y)\downarrow . \end{cases}$$

A função composta é denotada por $f = h \circ g$.

- O valor da função composta para x é $f(x) = h(g(x))$.
- Definição acima implica que a composição de funções totais produz uma função total.

Composição de Funções

- Composição de funções $\Rightarrow$ execução sequencial de TMs.
- Composição de funções computáveis (*Turing computable*) produz uma função computável.
- Argumento construtivo: combinação de TMs e macros vistas anteriormente.
- Generalizado pelo teorema abaixo.

Teorema 9.4.3 (Sudkamp)

A funções computáveis são fechadas sob a operação de composição.

Composição de Funções

- Teorema anterior pode ser usado para mostrar que uma função f é computável sem a necessidade de se mostrar explicitamente uma TM que computa f.
- $\Rightarrow$ Mostrar que f é uma composição de funções computáveis.

Exemplo 9.4.2 (Sudkamp)

As funções constantes de k-argumentos $c_i^{(k)}$ cujos valores são dados por $c_i^{(k)}(n_1, \ldots, n_k) = i$ são computáveis.
A função $c_i^{(k)}$ pode ser definida como

$$c_i^{(k)} = \underbrace{s \circ s \circ \cdots \circ s}_{i \text{ vezes}} \circ z \circ p_1^{(k)}$$

aonde cada função individual já foi vista como computável.

Funções Não-Computáveis

- Função f é computável $\Rightarrow$ existe uma TM que computa f.
- Existem funções numéricas que não são computáveis!
- Como mostrar isso?
- Na Aula 00, argumento de diagonalização mostrou que o conjunto de funções numéricas unárias totais é incontável.
- Se o conjunto de funções numéricas unárias totais e computáveis não for incontável então há mais funções do que máquinas que as computam.
- Esse é exatamente o resultado do teorema abaixo.

Teorema 9.5.1 (Sudkamp)

O conjunto de funções numéricas unárias totais computáveis é infinito contável (enumerável).

Funções Não-Computáveis

- Como provar o teorema anterior?
- Uma TM é completamente determinada pela sua função de transição.
- Qualquer TM que computa uma função pode ser formada pelos componentes:
  1. Os estados da TM são um subconjunto de $Q_0 = \{q_i \mid i \geq 0\}$. ($Q_0$ é contável. Por quê?)
  2. $\Sigma = \{1\}$. (Σ é contável. Por quê?)
  3. O alfabeto da fita é um subconjunto de $\Gamma_0 = \{B, 1, X_i \mid i \geq 0\}$. ($\Gamma_0$ é contável. Por quê?)
- Uma transição de qualquer TM é um elemento do conjunto

$$T = Q_0 \times \Gamma_0 \times \Gamma_0 \times \{L, R\} \times Q_0.$$

- T é contável. Por quê?
- $\Rightarrow$ O produto Cartesiano de conjuntos contáveis também é contável.

Funções Não-Computáveis

- Uma TM é completamente determinada pela sua função de transição δ.
- Função δ é sempre um conjunto finito e portanto contável.
- $\Rightarrow \delta \subset \mathsf{T}$ é um conjunto próprio de T.
- Seja $\mathcal{M}_i = \{\delta \subset \mathsf{T} \mid \mathit{card}(\delta) \leq i\}$ o conjunto de todas as TMs de tamanho i.
- Para qualquer i, $\mathcal{M}_i$ é contável.
- Seja $k \in \mathbf{N}$. O valor k é arbitrário mas necessariamente finito.
- O conjunto $\mathcal{M} = \bigcup_{i=0}^{k} \mathcal{M}_i$ descreve todas as TM que podem ser criadas. (Basta tomar k suficientemente grande.)
- $\mathcal{M}$ é contável porque a união de conjuntos contáveis também é contável.

Funções Não-Computáveis

- O número de TMs distintas que podem ser criadas é infinito contável.
- O número de funções numéricas é incontável (isto é, "maior" que infinito contável).
- $\Rightarrow$ Há funções numéricas que não podem ser computadas por nenhuma TM.
- Exemplo 1: *halting problem* (Módulo 04).
- Exemplo 2: *busy beaver problem* (a seguir).

Busy Beaver Problem

- Tibor Radó (1962): *"On Non-Computable Functions"*.
- *Busy Beaver Problem*: Qual é o maior número finito de "1s" que podem ser produzidos em uma fita inicialmente vazia por uma TM com n estados?
- Problema facilmente expressável como uma função.
- Seja $BB(n)$ a maior quantidade de "1s" produzida por uma máquina com n estados.
- Para valores bem pequenos de n é fácil determinar os valores de BB:

$$BB(1) = 1 \qquad BB(2) = 4 \qquad BB(3) = 6$$

- Para valores maiores de n a situação é bem mais complicada.
  - Provar que $BB(4) = 13$ foi uma tese de doutorado.
  - Para $n > 4$ não se sabe os valores exatos de BB.

Busy Beaver Problem

- A função $BB : \mathbf{N} \rightarrow \mathbf{N}$ certamente é uma função numérica, embora não se conheça uma fórmula para ela.
- Observações sobre a função BB:
  1. $BB(n)$ é uma função bem definida. Ela existe. Para qualquer número de estados n, o número de TMs possíveis é finito.
  2. $BB(n)$ é estritamente crescente: para cada estado a mais sempre é possível escrever pelo menos um "1" a mais.
- A função BB pode ser computada por uma TM?
- Em outras palavras, existe uma TM M_{BB} que recebe $\overline{n}$ como entrada e retorna $\overline{BB(n)}$ como saída?
- A resposta para essa pergunta é não! A função BB não é computável.
- A prova é por contradição, similar aos argumentos vistos na Aula 00. (Não será apresentada aqui.)
- Assista aos vídeos no AVA se quiser saber mais.

Aula 01 – Máquinas de Turing e Funções Computáveis

Prof. Eduardo Zambon

Departamento de Informática (DI)
Centro Tecnológico (CT)
Universidade Federal do Espírito Santo (Ufes)

Algoritmos e Fundamentos da Teoria de Computação (ToCE)
Engenharia de Computação